JULY/AUGUST '88
Vol. 4 No. 5
$3.00 U.S. Funds

Roy Wisti Sep/88
275-B Sheldon Rd
Voluntown, CT 06384

BULK RATE
U.S. POSTAGE
PAID
COLTON, OR 97017
PERMIT #51

POSTMASTER:
PLEASE EXPEDITE DELIVERY
DATED MATERIAL
ADDRESS CORRECTION REQUESTED

The SINCLAIR Computer Technology Magazine

TIME DESIGN

MASS STORAGE!

(716) 834-1716

T & C SERVICES
20 LIBERTY TERRACE
BUFFALO, N.Y. 14215

(716) 834-1716

Call or write for a free catalog of products for the Timex Computer

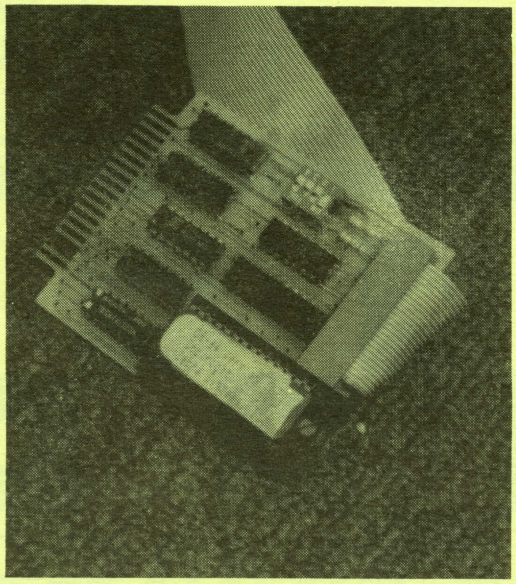
FOOTE SOFTWARE

The FOOTE PRINT PRINTER INTERFACE

- for Centronics parallel printers
- works in both 2068 and Spectrum mode
- compatible with OS-64 & Spectrum emulators
- EPROM socket and on/off switch on board
- works with **both** Tasman and Aerco driver software
- plugs into cartridge dock—door completely closes with cable running back under computer
- frees up rear edge connector allowing other peripherals to be used; less chance of a crash
- print driver software for LPRINT, LLIST, and COPY included for 2068 and Spectrum modes

FootePrint Interface w/software & cable $39.95
FootePrint with OS-64 option included $60.00
Bare board & instructions only $15.00
Cable only for use with bare board $15.00
Zero Insertion Force Socket option add $8.00

Summer Westcoast TS Fair Special



SOFTWARE	TS2068	TS1000	
Badgammon (Backgammon)	$10.00		$5.00
Advanced Math (Calculus)	$10.00		
Calorie Counter	$8.00		
U.S.A. (Pres. & States & Caps.)	$8.00		
Gambler (poker)	$8.00		
CHR$ (char. & graphics generator)	$10.00		
Hangman & TIC–TAC–TOE	$5.00		

FOOTE SOFTWARE
P. O. Box 14655
Gainesville, FL 32604
(904) 338-1273 (9AM-6PM EDT)
All prices are pre-paid and include shipping charges.
Florida residents must add state sales tax.

The Best of SUM

Some sample articles include: Building Your Own Spectrum Emulator, Repairing Your TS-1000, Word Processing Reviews for the 2068, UDGs on the TS-1000, Extensive Review of the Zebra Disk System, Adding a Keyboard to the 2068, and Enhancing the A & J Microdrive. 112 pages

$10.00

The Best of SUM, Part II

Articles include Building an EPROM Programmer, Sprites on the 2068, Adding RGB to 2068, QL Word Processing, What's Available for TS-1000, and much more. 60 pages

$7.00

both for $15.00

SEE THE *TIME DESIGNS* AND *FOOTE SOFTWARE* BOOTHS
AT THE NORTHWEST AND MID-WEST TIMEX SINCLAIR SHOWS!

TIME DESIGNS

The SINCLAIR Computer Technology Magazine

JULY/AUGUST '88

Time Designs Magazine Company
29722 Hult Road
Colton, Oregon 97017
USA
(503) 824-2658
CompuServe ID# 71350,3230

TIME DESIGNS MAGAZINE is published bi-monthly and is Copyright 1988 by the Time Designs Magazine Company, Colton, Oregon 97017. All rights reserved. Reproduction of this magazine in whole or in part by any means without written permission is prohibited by law.

SUBSCRIPTIONS: $16.95 a year for six issues (U.S. funds only), mailed in the U.S. All other countries please write for information on surface and air mail rates.

CUSTOMER SERVICE: Customer satisfaction is our goal. For subscription service problems, or any questions and comments, please write or call.

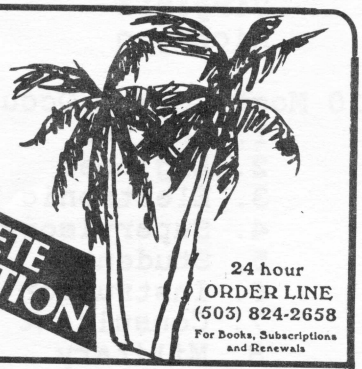
CHANGE OF ADDRESS: Please call or write our office if there is any change in your current mailing address to prevent delay or even loss of service.

RENEWAL TIME? To determine your expiration date, simply read the date posted in the upper-right corner of your mailing label (magazine cover). For an example, "Nov/89" would indicate that the November/December 1989 issue would be the last issue received. A form is provided elsewhere to renew your subscription. We also send one notice in case you forget. An early renewal is very much appreciated, and let us know we are doing an adequate job.

NOTICE: Contributors to TIME DESIGNS are independent of the TIME DESIGNS MAGAZINE CO., and opinions expressed in the contents of this publication are not necessarily those of the management staff or its advertisers. Time Designs Magazine Co. will not be held liable for any damage or consequences resulting from instructions, assertions of fact, review of products or companies provided in the magazine's content. It is recommended that anyone attempting to modify their computer or constructing an electrical project should seek help from more knowledgeable individuals.

SUMMER SPECIALS

BACK ISSUES



COMPLETE YOUR COLLECTION

TIME DESIGNS
*** VOLUME TWO CLOSEOUT ***
(limited quantity in stock)
$2.50 each, or 3 issues for only $6.00

NOV/DEC '85 (Vol.1 No.1)
JAN/FEB '86 (Vol.2 No.2) SOLD OUT
MAR/APR '86 (Vol.2 No.3)
MAY/JUN '86 (Vol.2 No.4)
JUL/AUG '86 (Vol.2 No.5)
SEP/OCT '86 (Vol.2 No.6)

*** VOLUME THREE ISSUES ***
(plenty of these in stock)
$3.00 each, or 3 issues for only $8.00

NOV/DEC '86 (Vol.3 No.1)
JAN/FEB '87 (Vol.3 No.2)
MAR/APR '87 (Vol.3 No.3)
MAY/JUN '87 (Vol.3 No.4)
JUL/AUG '87 (Vol.3 No.5)
SEP/OCT '87 (Vol.3 No.6)

24 hour ORDER LINE
(503) 824-2658
For Books, Subscriptions and Renewals

VISA and MASTERCARD ACCEPTED

TIME DESIGNS MAGAZINE CO.
29722 Hult Rd., Colton, OR 97017

Sinclair Survivalist HANDBOOK

The Sinclair Survivalist Handbook is a new 70 page book that is filled with previously un-published program listings and articles, written by regular contributor's to TIME DESIGNS, for the TS1000/ZX81, TS1500, TS2068, and the Sinclair QL. Examples include: "Adapting external keyboards to your TS1000", "BASIC Line Delete Utility", "Strategic Football", "Fix Your TS2068 Space Bar", "OS-64 Utilities", "Little League Scheduler", "Homemade ROM-Switch", "Draw Poker", "QL Super-BASIC Tutorial", "Using Quill With The QL Printer", "Inside the QL", and much more! If you like TIME DESIGNS...you'll like this new book. Order your copy today!

9.95

READER SURVEY RESULTS

Part One

Over 280 TIME DESIGNS readers responded to our recent survey. Which is approximately 11 percent of our circulation. Some of the results were most interesting. The rest of the results will conclude next issue. We sincerely hope that this data will provide developers of hardware and software, and the Timex Sinclair dealers with a better understanding of our TS user community.

Average Age of TDM Subscriber: 45
Oldest Reported Age: 79
Youngest Reported Age: 22

Male/Female Ratio: 22 Male/1 Female

States of Highest TS User Concentration:
- California
- New York
- Pennsylvannia
- Florida
- Texas
- Ohio
- Illinois
- Virginia
- Michigan

10 Most Common Occupations of TS Users:
1. Retired
2. Engineer
3. Electronic Technician
4. Supervisory/Management
5. Student
6. Instructor
7. Consultant
8. Military
9. Sales
10. Librarian

Top 5 Hobbies of TS Users:
1. Computers
2. Electronics
3. Photography
4. Amateur Astronomy
5. Gardening

Computer Used The Most:
1. TS2068
2. TS1000
3. QL
4. IBM (or compatible)
5. Commodore 64

Other Computer Owned:
1. TS1000
2. TS2068
3. ZX80
4. TS1500
5. QL

Most Popular Printer:
1. 2040 Thermal Printer
2. Gorilla Banana
3. Epson RX-80
4. Star NX-10
5. Panasonic KXP-1080

Most Popular Mass Storage Device Used:
1. Cassette Tape
2. Floppy Disk
3. Microdrive Cartridges/Wafers
4. EPROMs

Most Popular TS2068 Disk Systems:
1. Aerco FD-68
2. Larken LKDOS
3. Zebra FDD
4. Oliger SAFE DOS
5. Ramex MK

5 Most Popular Monitor Devices:
1. Color and B/W TV
2. Magnavox RGB
3. Zenith
4. Sanyo
5. Commodore

TIMEX SINCLAIR NEWS

If only TIME DESIGNS were printed in color! Then we could show you what the actual "Sir Clive" screen dump (above) really looks like. It was produced using a program called THE ARTIST, a TS2068 computer, an OKIMATE 20 Color Printer, and a special interface and printing utility software.

To back-track just a bit...a couple of years ago, a puzzled TDM subscriber sent a letter to the editor, and posed the question whether the OKIMATE 20 could be used with the 2068. Seems that the Okimate was configured to operate with "most popular brands" of personal computers, with optional "Plug 'n Print" interface/software packages...all of the brands, except the Timex (of course).

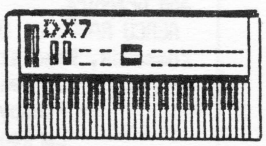
Thanks to the research and development of John McMichael (who also adapted an inexpensive Commodore plotter to the 2068), Timex users no longer have to face discrimination. Using the Okimate 20, the Commodore "Plug 'N Print" package, and John's new "Commodore serial port emulation circuit board", high quality screen dumps can now be enjoyed in color. John also offers several print utility programs to help get the job done.

Not only is the Okimate 20 a good color printer, but it is also suited for normal printer use (including NLQ mode).

Information about the interface, and related 2068 software can be obtained by sending a legal SASE to: John McMichael, 1710 Palmer Drive, Laramie, WY 82070.

The Okimate 20 must be purchased elsewhere, and is available everywhere. Try Sears, Target, Best, or Lyco Computer Inc. (1-800-233-8760). Typical discount price for printer and "Plug 'n Print" package is right around $180.

MIDI FOR THE 2068



Another frequently requested device for the TS2068 is a MIDI Interface. MIDI stands for "Musical Instrument Digital Interface". Which is the means for hooking up electronic synthesizer keyboards, drum boxes (and much more) to your computer. Other computer brands like the ATARI ST and IBM PC are very strong in the area of MIDI support, both hardware and software.

MIDI is an invaluable tool for music students, composers, and live performers. And since MIDI is a word-wide industrial standard among electronic music instrument manufacturers and computer manufacturers, it wouldn't matter if you had an ATARI or a SINCLAIR, the hardware compatibility should be the same.

Recently, Richard Hurd, a TDM subscriber and occasional contributor wrote, "I have had success implementing MIDI on my TS2068. I also would be happy to hear from anyone interested in this."

Richard has purchased RAM Electronic's MUSIC MACHINE, a MIDI interface for the Spectrum, from England (see review in the November '86 issue of ZX Computing), and also some accompanying MIDI software from a company called QUASAR. To operate the Spectrum hardware and software on the TS2068, Richard purchased John Mathewson's "Twister Board" for the rear expansion bus, and also used a Spectrum emulator.

For further details, addresses, and even tips on ordering from Great Britain, send a SASE to: Richard Hurd, PO Box 153, Warrenton, OR 97146.

WHERE GOEST FRED??

Fred Nachbaur, formally of Nelson, British Columbia, Canada, and highly-respected authority on Sinclair computers, has taken several new turns. Most recently, he has accepted a position with a firm in Ottawa, and will be turning his TS1000/ZX81 product line over to other Timex Sinclair dealers.

Fred's own company, Silicon Mountain Computers, will be renamed "Silicon Mountain Electronics", which, as the name implies, will pursue avenues of a more general electronic nature as well as computers.

As a former TS software/hardware producer and supplier, Fred found that he lacked critical time and funding for development of various special projects, including one particular project...a new type of computer.

Fred recently told TIME DESIGNS that, "It should be clearly understood, however, that this project is by no means a certainty at this point. It's not because of the infamous "big IF", rather it depends on a whole lot of "little if's". I have carefully chosen a core of potential developers who have expressed an interest to investigate the potentials; IF we all agree on the route to take; IF we all find the time to do our parts; IF the economics fall into place; IF the result of our brain-pooling results in a marketable product; IF no one comes up with a better mouse before we build a better trap...then there will be a new computer. But don't believe anything you hear, unless you hear it from us. If it does happen, it will not be, as rumour has it, a Timex "clone". The new machine will have some common features, such as elegance in simplicity, but a new machine in its own right."

Fred wants everyone to know that he will continue to be involved with the ZX81 family of fine computers; as a user, writer, and hacker, but not as a commercial supplier of TS software.

SECRET STUFF

Nigel Searle, a close associate of Sir Clive Sinclair for over sixteen years, announced to the General Assembly of the Boston Computer Society, on

June 22, that Sir Clive was involved in developing some highly secretive computer equipment, and that he (Sir Clive) would personally announce detailed plans of the project and launch it world-wide at an upcoming BCS meeting.

The only speculation and possible clues about the new computer equipment are coming out of the British press. Supposedly, Sir Clive is developing a new desktop computer based on transputer chip technology, similar to the INMOS transputer, only Sir Clive felt that the INMOS was unsuitable for his project, and went out and developed his own transputer. The new desktop will reportedly outperform any PC technology currently available, processing data more than 10 times faster than an IBM AT. The new machine will be marketed under the CAMBRIDGE COMPUTER LTD label, just as the Z88 Laptop is.

CLEVELAND

Saturday, August 27 and Sunday, August 28, marks the date for the upcoming MIDWEST SINCLAIR COMPUTER CONFERENCE, which will be held at the Beck Center in Lakewood, Ohio (a suburb of Cleveland).

The Conference will feature TS exhibitors like Zebra Systems Inc., Sharp's Inc., Time Designs, and others; as well as seminars by Bill Ferrebee, James DuPuy, Basil Wentworth, Dave Hoshor, Thomas Simon, and others.

For complete details, info on accomodations available in the area, pre-registration forms, and more, send a SASE to: Andy Kosiorek, 2192 Glenbury Ave., Lakewood, OH 44107. For alternate information contact: James DuPuy, 6514 Bradley Ave (down), Parma, OH 44129, (216) 661-4105.

If you live in Ohio, any of the surrounding states, the midwest proper, Ontario (Canada), most anywhere on the eastcoast and southern states (or anywhere!)...come to the show and exchange ideas and information with fellow Timex Sinclair users.

NEW RELEASES

PODNUH is a clever name for a new TS2068 program, which has been thoroughly tested since its conception in 1986. This Machine Code program includes a BASIC programmer (called "Supra-Basic") with a swift and reliable method of passing parameters to, and calling other Machine Code programs. These "other" programs may be customized routines, extensions of BASIC, utility programs, or complete applications programs. A PODNUH (version 1) package is available for $17.00 + $2.00 postage, which includes selectable type fonts, a perpetual calendar, note pad, scientific calculator, and more. "Add-On" options will continue to be added, such as WYNN DOE (a windowing utility) for $5.00. The author is also interested in sharing his program with programmers or user groups, and is offering a disassembled listing with documentation for $1.00, with the hope that PODNUH is adopted as a new 2068 standard. Send check or money order to: Ron Ruegg, 37529 Perkins Road, Prairieville, LA 70769.

Many Timex fans have heard or read about the research that William J. Pederson of THE WIDJUP CO. has conducted on the TS2068 ROM/Operating System, from articles in several user group newsletters and magazines. Now there is a 160 page book by Mr. Pederson called "TOURING THE TS2068 ROM OPERATING SYSTEM". While this type of book isn't for everyone, it will be of interest to programmers and 2068 hacker-types everywhere. It contains a complete annotated disassembly of the Home ROM and the XROM, along with several other tables of data. Mr. Pederson offers a theory which one may or might not accept about the development of the Timex ROM and bank-switching routines...but it all makes for interesting reading. The book is priced at $20.00 + $3.00 postage and is available from: The WIDJUP Co., 1120 Merrifield S.E., Grand Rapids, MI 49507.

Arnold Ramaker, PO Box 263, Plymouth, WI 53073, (414) 893-8865, is busy designing an expansion box for the ZX81, TS1000, TS1500, TS2068, and Sinclair QL computers. Any one of the computers can be placed inside the supplied case. The expansion box will feature multi-expansion slots and provisions for attaching a monitor, and several other peripherals. Mr. Ramaker would like to hear from folks who are interested in purchasing an expansion box like this, to get an idea on what price range and any additional features people would like to see incorporated.

Matthew Zenkar, 142 Holcroft Rd., Rochester, NY 14612, (716) 663-2048, is offering a utility program which will allow QL owners who use the Digital Precision Desktop Publisher software package, to dump their files to Hewlett Packard-compatible laser printers. Write for information and price.

The S.A.I.N. (Sinclair Artificial Intelligence Network) special interest group is now forming. It is for any Sinclair user interested in A.I., Micro-PROLOG, LISP, and other related topics. For further information, send a SASE to: Pete Fischer, PO Box 2002, Tempe, AZ 85281, or call the TIMEWARP BBS, (617) 481-0555 (setting: 8/1/N, 300 baud).

LARKEN PRESENTS ...

UP TO 256K RAM for your 2068

- Expand your 2068 with up to 256K of battery backed up Ram
- Larken Operating system lets you SAVE to memory, just like cassette or disk. (Floppy disk not required)
- All Cassette commands supported. Very Fast and Reliable .
- Can be used with ALL existing 2068 or Spectrum software.
- Uses the new 32K static ram chips, 62256LP or 43256LP
- System consists of Larken Cartridge and Rear Memory Board.

** PRICE - MEMORY SYSTEM with 64K Ram $129.00
- MEMORY SYSTEM with 0 K ..... $ 95.00

LARKEN 2068 FLOPPY DISK SYSTEM

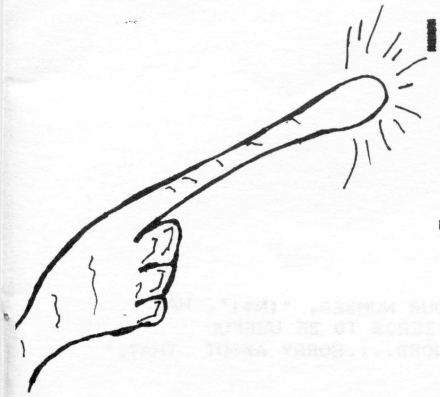
- The most advanced Dos available for the 2068/Spectrum . LKdos uses ALL Commands such as CAT MERGE ERASE LOAD SAVE PRINT OPEN etc. Also can support RAMDISK up to 256K and Sequential / Random Access Files (with additional software). The Larken Disk Interface can handle up to 4 floppys for up to 3.2 MegaBytes of storage. Also NMI Snapshot Save Button and KEMPSTON Joystick port on interface Also 10 Extended Basic commands for Windows and Graphics.

AERCO RAMEX or OLIGER Disk users can add LKdos for more commands, Ramdisk and access to all LKdos software

** PRICE - Larken Floppy Disk System $119.95
- Floppy Disk IF with 0 K Memory board .. $169.95
- Larken Disk Editor ..... $ 15.00
- Sequential/Random access files ..... $ 15.00
- Xmodem to Disk Modem package ..... $ 15.00
- ZX-81 Floppy Interface (15 left) ... $ 99.95
- LKDOS for Aerco,Ramex or Oliger Disk IF $ 59.95

(All prices are US , Add 6$ Shipping)

LARKEN ELECTRONICS RR#2 NAVAN ONTARIO CANADA K4B-1H9
(613)-835-2680



"ZX...PHONE HOME!"

A TS1000/TS1500/ZX81 Program

To Help Remember Telephone Numbers

by David Hartman

The following program develops words based on the last four digits of telephone numbers. I have found that a phone number is remembered as a word, easier than the four numbers.

This program will run on a TS1000 or ZX81 with a 16K RAM pack installed, or on a TS1500. This program should also work with a TS2068, with only minimal changes to the program.

Those users with a 16K TS1000/ZX81, should end up with "13505" printed on the screen, after entering the following line: PRINT (PEEK 16388 + 256 * PEEK 16389) - (PEEK 16396 + 256 * PEEK 16397)

If "13505" is your answer, then you have most likely typed in the program listing correctly.

```
Line Uses:

1-24: Opening screen
25-40: Instructions and input
50-70: Check for '1' and '0'
80: See 8000
110-200: Assemble words by slicing
202-255: Print results on screen
260-310: Copy and continue
500-575: Assemble words if a '1' or '0' are involved
600-640: Re-establish phone number for printing
5010-5040: Initialize, set up arrays
8000-8040: Not useful message

VARIABLES:

n$: Holds entered phone number
w$: Holds all possible letter combinations
l$: Holds telephone dial information
f: Flag indicates if a '1' or a '0' is in the number
(Important to know because 1 and 0 do not have assigned letters)
x: Increment control
j, k, l, m: For/Next control
```

```
  1 REM WORDS FROM TELEPHONE NUMBERS
  2 REM 105/1.1 3/20/88
  7 SLOW
  8 CLS
 10 PRINT AT 6,0;"WORDS FROM TELEPHONE NUMBERS";AT 14,0;"LETS SEE WHAT YOUR NUM
BER SPELLS"
 15 PRINT AT 21,0;"(C) 1988 DAVID HARTMAN"
 20 GOSUB 5000
 22 FOR J=1 TO 60
 24 NEXT J
 25 CLS
 27 PRINT TAB 8;"TELEPHONE WORDS",,,
 30 PRINT "ENTER THE LAST FOUR DIGITS OF  YOUR TELEPHONE NUMBER TO SEE","WHAT
THEY SPELL. THEN, USE THAT WORD WHEN SOMEBODY WANTS YOUR","NUMBER. THEY PROBABLY
 WILL BE","ABLE TO REMEMBER IT WITHOUT","WRITING IT DOWN."
 40 INPUT N$
 43 IF LEN N$<4 THEN GOTO 90
 45 LET F=0
 50 FOR J=1 TO 4
 65 IF N$(J)="1" OR N$(J)="0" THEN LET F=F+1
 67 IF N$(J)="1" THEN LET N$(J)="Q"
 68 IF N$(J)="0" THEN LET N$(J)="Z"
 70 NEXT J
 80 IF F>1 THEN GOTO 8000
 85 GOTO 110
 90 PRINT AT 15,0;"YOU DID NOT ENTER A 4 DIGIT",
    "NUMBER. PLEASE TRY AGAIN."
 95 FOR J=1 TO 60
 97 NEXT J
100 GOTO 25
110 LET X=1
115 FAST
120 FOR J=1 TO 3
130 FOR K=1 TO 3
140 FOR L=1 TO 3
150 FOR M=1 TO 3
153 IF F THEN GOTO 500
167 LET W$(X)=L$(VAL N$(1),J)+L$(VAL N$(2),
    K)+L$(VAL N$(3),L)+L$(VAL N$(4),M)
169 LET X=X+1
170 NEXT M
180 NEXT L
190 NEXT K
200 NEXT J
202 CLS
203 SLOW
205 GOSUB 600
210 PRINT "HERE ARE THE CHOICES FOR ";N$;":"
212 PRINT
215 LET J=1
220 FOR K=J TO J+5
225 IF K=82 THEN GOTO 260
230 PRINT W$(K);".";
240 NEXT K
245 PRINT
250 LET J=K
255 GOTO 220
260 PRINT AT 21,0;"(C) COPY?"
270 INPUT A$
280 IF A$<>"C" THEN GOTO 25
290 PRINT AT 21,0;"                "
300 COPY
310 RUN
500 IF N$(1)="Z" THEN LET W$(X)="0"+L$(VAL N$
    (2),K)+L$(VAL N$(3),L)+L$(VAL N$(4),M)
515 IF N$(1)="Q" THEN LET W$(X)="1"+L$(VAL N$
    (2),K)+L$(VAL N$(3),L)+L$(VAL N$(4),M)
520 IF N$(2)="Z" THEN LET W$(X)=L$(VAL N$(1),
    J)+"0"+L$(VAL N$(3),L)+L$(VAL N$(4),M)
```

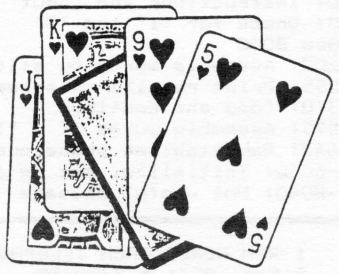
```
 530 IF N$(2)="Q" THEN LET W$(X)=L$(VAL N$(1),
     J)+"1"+L$(VAL N$(3),L)+L$(VAL N$(4),M)
 540 IF N$(3)="Z" THEN LET W$(X)=L$(VAL N$(1),
     J)+L$(VAL N$(2),K)+"0"+L$(VAL N$(4),M)
 550 IF N$(3)="Q" THEN LET W$(X)=L$(VAL N$(1),
     J)+L$(VAL N$(2),K)+"1"+L$(VAL N$(4),M)
 560 IF N$(4)="Z" THEN LET W$(X)=L$(VAL N$(1),
     J)+L$(VAL N$(2),K)+L$(VAL N$(3),L) +"0"
 570 IF N$(4)="Q" THEN LET W$(X)=L$(VAL N$(1),
     J)+L$(VAL N$(2),K)+L$(VAL N$(3),L) +"1"
 575 GOTO 169
 580 STOP
 600 FOR J=1 TO 4
 610 IF N$(J)="Q" THEN LET N$(J)="1"
 620 IF N$(J)="Z" THEN LET N$(J)="0"
 630 NEXT J
 640 RETURN
5010 DIM L$(10,3)
5015 DIM W$(81,4)
```

```
5020 LET L$(1)=""
5022 LET L$(2)="ABC"
5024 LET L$(3)="DEF"
5026 LET L$(4)="GHI"
5028 LET L$(5)="JKL"
5030 LET L$(6)="MNO"
5032 LET L$(7)="PRS"
5034 LET L$(8)="TUV"
5036 LET L$(9)="WXY"
5038 LET L$(10)=""
5040 RETURN
8000 CLS
8005 GOSUB 600
8010 PRINT AT 14,0;"YOUR NUMBER, ";N$;", HAS
     TOO MANY ONES OR ZEROS TO BE USEFUL
     IN  CREATING A WORD....SORRY ABOUT  THAT."
8020 FOR J=1 TO 100
8030 NEXT J
8040 GOTO 25
8999 STOP
9000 SAVE "TELEWORDS"
9010 RUN
```

STUD POKER

A Casino-Style Game Listing For The TS2068

by William C. Andrews



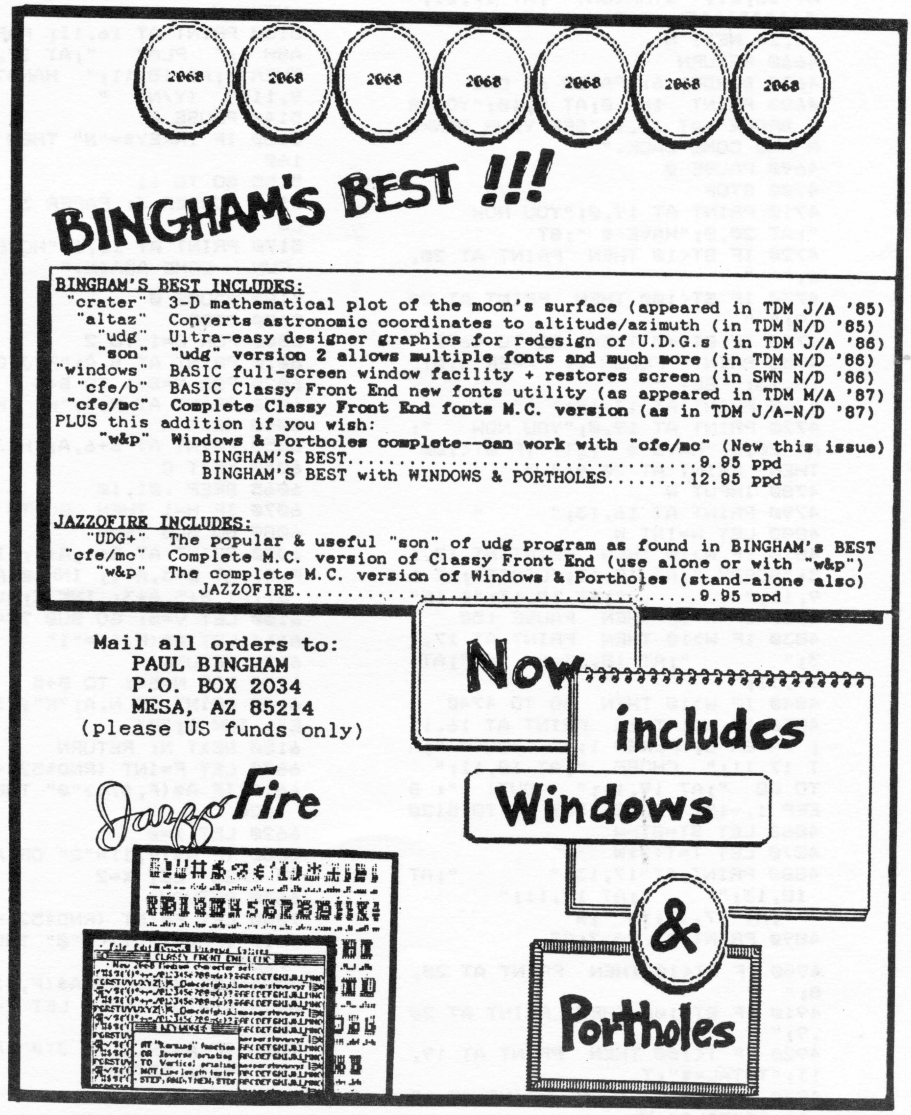
This is a TS 2068 program to play FIVE CARD STUD POKER against the computer, the dealer. After an ante you bet on each card dealt. The dealer matches your bet. You may go out at any time by betting "0". You must pay to see the hole card if needed. The dealer is also the banker and will keep track of the game's progress.

When typing the program please note that letters in quotes in lines 6010, 6030, 6050, 6140, 7020, 9991 and the last letter in lines 8001 to 8052 are in GRAPHIC mode for UDG's. Line 9992 clears color from the screen for working on the program. Line 6000 selects cards randomly and line 6110 prevents duplication. Cards are shuffled for each game.

For a tape of this program send $ 10.00 pp to me at 30 oak Knoll Drive, San Anselmo, CA. 94960.

```
  10 LET ST=100
  15 IF ST<=0 THEN  GO TO 4670
  20 BORDER 4: PAPER 4: CLS : LE
T T=0: GO SUB 8000
  25 PAPER 7: FOR N=2 TO 14: PRI
NT AT N,0;"
          ": NEXT N
  30 LET PR=0: LET DR=0: LET PL=
0: LET DL=0: DIM X$(6): DIM Y$(6
)
  35 RANDOMIZE
  40 PRINT  PAPER 4;"      "; PAP
ER 2; INK 9; BRIGHT 1;AT 0,5;" F
IVE CARD STUD POKER "; BRIGHT 0;
 PAPER 4;"      "
  45 INK 1
  50 LET K$="▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀
▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀
```

```
1190 IF C<=D THEN  GO TO 1220
1200 LET A(M)=D
1210 LET A(M+1)=C
1220 NEXT M
1230 NEXT N
1240 FOR N=2 TO 5
1250 IF A(N)=A(N-1)+1 THEN  GO T
O 1270
1260 return
1270 NEXT N
1280 LET STR=1: RETURN
3999 STOP
4000 IF P(2)=P(1) THEN  LET PL=P
(2): GO TO 4150
4010 RETURN
4020 IF P(3)=P(1) OR P(3)=P(2) T
HEN  LET PL=P(3): GO TO 4150
4021 RETURN
4030 IF PR=1 THEN  IF P(3)=P(1)
AND P(3)=P(2) THEN  LET PL=P(3):
 GO TO 4190
4040 RETURN
4050 IF PR=1 THEN  IF P(4)=P(1)
AND P(4)=P(2) OR P(4)=P(1) AND P
(4)=P(3) OR P(4)=P(2) AND P(4)=P
(3) THEN  LET PL=P(4): GO TO 419
0
4060 IF P(4)=P(1) OR P(4)=P(2) O
R P(4)=P(3) THEN  LET PL=P(4): G
O TO 4150
4070 RETURN
4080 IF PR=3 THEN  IF P(5)=P(1)
AND P(5)<>P(4) OR P(5)=P(2) AND
P(5)<>P(4) OR P(5)=P(3) AND P(5)
<>P(4) OR P(5)=P(4) AND P(5)<>P(
3) THEN  LET PL=P(5): GO TO 4200
4090 IF PR=3 THEN  IF P(5)=P(2)
OR P(5)=P(3) OR P(5)=P(4) THEN
LET PL=P(5): GO TO 4200
4100 IF PR=2 THEN  IF P(5)=P(1)
AND P(5)=P(2) OR P(5)=P(1) AND P
(5)=P(3) OR P(5)=P(1) AND P(5)
=P(4) OR P(5)=P(2) AND P
(5)=P(3) OR P(5)=P(2) AND P(5)
=P(4) OR P(5)=P(3) AND P(5)=P(4)
 THEN  LET PL=P(5): GO TO 4190
4110 IF PR=1 THEN  IF P(5)=P(1)
AND P(5)=P(2) OR P(5)=P(1) AND
 P(5)=P(3) OR P(5)=P(1) AND P(5)
=P(4) OR P(5)=P(2) AND P(5)=
P(3) OR P(5)=P(2) AND P(5)=P(
4) OR P(5)=P(3) AND P(5)=P(
4) THEN  LET PL=P(5): GO TO 4190
4120 IF P(5)=P(1) OR P(5)=P(2)
 OR P(5)=P(3) OR P(5)=P(4) THE
N  LET PL=P(5): GO TO 4150
4140 RETURN
4150 LET PR=PR+1
4160 IF PR=1 THEN  PRINT  INK 2;
AT 16,2;"ONE PAIR"; INK 1;AT 17,
2;"▪▪▪▪▪▪▪▪▪▪": BEEP .3,10
4170 IF PR=2 THEN  PRINT  INK 2;
     AT 16,2;"TWO": BEEP .3,10:
BEEP .3,10
4180 RETURN
4190 LET PR=PR+1
4200 LET PR=PR+1
4210 IF PR=5 THEN  GO TO 4260
4220 IF PR=3 THEN  PRINT  INK 2;
AT 16,1;"  THREE  ";AT 17,1;"OF
A KIND"; INK 1;AT 18,1;"▪▪▪▪▪▪▪▪▪
▪": BEEP .2,10: BEEP .2,10: BEEP
 .2,10
4230 IF PR=3 THEN  RETURN
4240 IF PR=4 THEN  PRINT  INK 2;
AT 16,1;"  FOUR   ";AT 17,1;"OF
A KIND"; INK 1;AT 18,1;"▪▪▪▪▪▪▪▪▪
▪": BEEP .2,10: BEEP .2,10: BEEP
 .2,10: BEEP .2,10
4250 RETURN
4260 IF PR=5 THEN  PRINT  INK 2;
AT 16,0;"FULL HOUSE"; INK 1;AT 1
7,0;"▪▪▪▪▪▪▪▪▪▪▪": BEEP .2,10: BE
EP .2,10: BEEP .2,10: BEEP .2,10
: BEEP .2,10: BEEP .2,10:
BEEP .2,10: BEEP .2,10: BEEP
 .2,10
4270 RETURN
4280 IF X$(1)=X$(2) THEN  IF X$(
2)=X$(3) THEN  IF X$(3)=X$(4) TH
EN  IF X$(4)=X$(5) THEN  LET PR=
6
4290 IF PR=6 THEN  PRINT  INK 2;
AT 16,1;"  FLUSH  "; INK 1;AT 17
,1;"▪▪▪▪▪▪▪▪▪▪": FOR N=1 TO 10: B
EEP .2,10: NEXT N
4291 GO SUB 1000: IF STR=1 THEN
 LET PR=7
4292 IF PR=7 THEN  PRINT  INK 2;
AT 16,1;"STRAIGHT "; INK 1;AT 17
,1;"▪▪▪▪▪▪▪▪▪▪": FOR N=1 TO 10: B
EEP .2,10: NEXT N
4300 RETURN
4320 IF D(3)=D(2) THEN  LET DL
=D(2): GO TO 4490
4321 RETURN
4323 IF DR=1 THEN  IF D(4)=D(2)
AND D(4)=D(3) THEN  LET DL=D(4):
 GO TO 4540
4324 IF D(4)=D(2) OR D(4)=D(3) T
HEN  LET DL=D(4): GO TO 4490
4325 RETURN
4326 IF DR=3 THEN  IF D(5)=D(2
) AND D(5)=D(3) AND D(5)=D(4) TH
EN  LET DL=D(5): GO TO 4540
4370 IF DR=1 THEN  IF D(5)=D
(2) AND D(5)=D(3) OR D(5)=D(2) A
ND D(5)=D(4) OR D(5)=D(3) AND D(
5)=D(4) THEN  LET DL=D(5): GO TO
 4540
4380 IF D(5)=D(2) OR D(5)=
D(3) OR D(59=D(4) THEN  LET
DL=D(5): GO TO 4490
4390 RETURN
4400 IF DR=3 THEN  IF D(1)=D(2
) AND D(1)<>D(5) OR D(1)=D(3) AN
D D(1)<>D(5) OR D(1)=D(4) AND
D(1)<>D(5) OR D(1)=D(5) AND D(1)
<>D(4) THEN  LET DL=D(1): GO TO
4540
4410 IF DR=2 THEN  IF D(1)=D(2)
OR D(1)=D(2) OR D(1)=D(3) THEN
  LET DL=D(1): GO TO 4530
4420 IF DR=1 THEN  IF D(1)=D(2)
AND D(2)=D(3) OR D(1)=D(2) AND
 D(2)=D(4) OR D(1)=D(2) AND
D(2)=D(5) OR D(1)=D(3) AND D
(3)=D(4) OR D(1)=D(3) AND D(3)
=D(5) OR D(1)=D(4) AND D(4)=
D(5) THEN  LET DL=D(1): GO TO 45
40
4430 IF DR=1 THEN  IF D(1)=D(2)
OR D(1)=D(3) OR D(1)=D(4) OR D(1
)=D(5) THEN  LET DL=D(1): GO TO
4460
4440 IF D(1)=D(2) OR D(1)=D(3) O
R D(1)=D(4) OR D(1)=D(5) THEN  L
ET DL=D(1): GO TO 4460
4450 RETURN
4460 LET DR=DR+1
4480 GO TO 4500
4490 LET DR=DR+1
4500 IF DR=1 THEN  PRINT  INK 2;
AT 16,21;"ONE PAIR"; INK 1;AT 17
,21;"▪▪▪▪▪▪▪▪▪▪": BEEP .3,2
4510 IF DR=2 THEN  PRINT  INK 2;
AT 16,21;"TWO": BEEP .3,2: BEEP
.3,2
```

```
4520 RETURN
4530 LET DR=DR+1
4540 LET DR=DR+1
4550 LET DR=DR+1
4560 IF DR=5 THEN  GO TO 4610
4570 IF DR=3 THEN  PRINT  INK 2;
AT 16,21;"  THREE  ";AT 17,21;"O
F A KIND"; INK 1;AT 18,21;"▚▚▚▚
▚▚▚▚": BEEP  .2,2: BEEP .2,2: BE
EP .2,2
4580 IF DR=3 THEN  RETURN
4590 IF DR=4 THEN  PRINT  INK 2;
AT 16,21;"  FOUR   ";AT 17,21;"O
F A KIND"; INK 1;AT 18,21;"▚▚▚▚
▚▚▚▚": BEEP .2,2: BEEP .2,2: BEE
P .2,2: BEEP .2,2
4600 RETURN
4610 IF DR=5 THEN  PRINT  INK 2;
AT 16,21;"FULL HOUSE";AT 17,21;"
▚▚▚▚▚▚▚▚▚▚▚▚": FOR N=1 TO 8: BEEP
.2,2: NEXT N
4620 RETURN

4630 IF Y$(1)=Y$(2) THEN  IF Y$(
2)=Y$(3) THEN  IF Y$(3)=Y$(4) TH
EN  IF Y$(4)=Y$(5) THEN  LET DR=
6
4640 IF DR=6 THEN  PRINT  INK 2;
AT 16,21;"  FLUSH  ";AT 17,21;"▚
▚▚▚▚▚▚▚▚▚": BEEP .2,2: BEEP .2,
2: BEEP .2,2: BEEP .2,2: B
EEP .2,2: BEEP .2,2: BEEP .2,2
4650 GO SUB 1030: IF STR=1 THEN
 LET DR=7
4655 IF DR=7 THEN  PRINT  INK 2;
AT 16,21;" STRAIGHT ";AT 17,21;"
▚▚▚▚▚▚▚▚▚▚▚▚": FOR N=1 TO 8: BEEP
.2,2: NEXT N
4660 RETURN
4670 BORDER 6: PAPER 6: CLS
4680 PRINT  INK 0;AT 9,10;"YOU'R
E BROKE";AT 11,1;"SEE YOUR BANKE
R AND COME BACK."
4690 PAUSE 0
4700 STOP
4710 PRINT AT 19,0;"YOU NOW
";AT 20,0;"HAVE-$ ";ST
4720 IF ST<10 THEN  PRINT AT 20,
8;"  "
4730 IF ST<100 THEN  PRINT AT 20
,9;" "
4740 IF ST<=0 THEN  GO TO 4670
4750 PRINT  FLASH 1; PAPER 6;AT
16,13;" BET "
4760 PRINT AT 17,15;"  "
4770 PRINT AT 19,0;"YOU NOW   ";
AT 20,0;"HAVE-$ ";ST: IF ST<100
THEN  PRINT AT 20,9;" "
4780 INPUT W
4790 PRINT AT 16,13;"      "
4800 LET W=INT W
4810 IF W>10 THEN  PRINT AT 17,1
3;"SORRY,";AT 18,13;"LIMIT";AT 1
9,11;"          ";AT 19,13;"$ 10"
4820 IF W>10 THEN  PAUSE 150
4830 IF W>10 THEN  PRINT AT 17,1
3;"       ";AT 18,13;"     ";AT
19,11;"          "
4840 IF W>10 THEN  GO TO 4740
4850 IF W<1 THEN  PRINT AT 16,11
; PAPER 5; FLASH 1;"   YOU   ";A
T 17,11;"  CHOSE  ";AT 18,11;"
TO GO  ";AT 19,11;"   OUT!  ": B
EEP 1,-15: PAUSE 150: GO TO 5120
4860 LET ST=ST-W
4870 LET T=T+2*W
4880 PRINT AT 17,13;"       ";AT
 18,13;"     ";AT 19,11;"
  ";AT 17,13;"$ ";W
4890 PRINT AT 20,7;ST

4900 IF ST<10 THEN  PRINT AT 20,
8;"  "
4910 IF ST<100 THEN  PRINT AT 20
,9;" "
4920 IF T<100 THEN  PRINT AT 19,
11;"TOTAL=$";T
4930 IF T>=100 THEN  PRINT AT 19
,11;"TOTAL$";T
4940 RETURN
4950 FOR I=16 TO 20
4960 PRINT AT I,10; PAPER 4;"▮";
AT I,20;"▮"
4970 NEXT I
4980 PRINT AT 15,10; PAPER 4;"▀
▀▀▀▀▀▀▀▀"
4985 FOR N=16 TO 19: PRINT AT N,
11; PAPER 7; INK 1;"          ";
NEXT N
4990 PRINT AT 20,10; PAPER 4;"▮"
; PAPER 0; INK 9;" THE POT "; PA
PER 4;"▮"
4995 RETURN
5010 IF PR>DR THEN  GO TO 5060
5015 IF DR>PR THEN  GO TO 5110
5020 IF PR=0 AND DR=0 THEN  GO T
O 5130
5030 IF PR=DR AND PL>DL THEN  GO
 TO 5060
5040 IF PR=DR AND DL>PL THEN  GO
 TO 5110
5050 GO TO 5130
5060 PAUSE 80
5070 PRINT  FLASH 1; INK 1;AT 16
,1;"  WINNER "; FLASH 0; INK 0;A
T 17,1;"          ";AT 17,3;"$ ";
T: FOR N=1 TO 8: BEEP .1,10: NEX
T N
5080 LET ST=ST+T
5090 PRINT AT 20,7;ST
5100 GO TO 5130
5110 PAUSE 80
5120 PRYNT  FLASH 1; INK 1;AT 16
,21;" WINNER  "; FLASH 0; INK 0;
AT 17,21;"          ";AT 17,23;"$
 ";T: FOR N=1 TO 8: BEEP .1,-10:
 NEXT N
5130 PRINT AT 16,11; PAPER 6; FL
ASH 1;"  PLAY   ";AT 17,11;" ANO
THER ";AT 18,11;"  HAND?  ";AT 1
9,11;"  (Y/N)  "
5140 PAUSE 0
5150 IF INKEY$="N" THEN  GO TO 5
160
5155 GO TO 11
5160 BORDER 3: PAPER 3: INK 7: C
LS
5170 PRINT AT 10,1;"HOPE YOU HAD
 FUN---COME AGAIN."

5180 PAUSE 0
5190 STOP
6000 FOR C=1 TO 2
6010 PRINT AT B,A;"FJJJG"
6020 FOR N=B+1 TO B+5
6030 PRINT AT N,A;"K   K"
6040 NEXT N
6050 PRINT AT B+6,A;"HJJJI"
6060 NEXT C
6065 BEEP .01,10
6070 IF H=1 THEN  GO TO 6130
6080 GO SUB 7000
6090 PRINT AT B+1,A+1; INK X;A$(
F,4);AT B+3,A+1; INK 0;A$(F,2 TO
 3);AT B+5,A+3; INK X;A$(F,4)
6100 LET V=0: GO SUB 7040
6110 LET A$(F,1)="1"
6120 RETURN
6130 FOR N=B+1 TO B+5
6140 PRINT AT N,A;"K"; INK 1;"EE
E"; INK 0;"K"
6150 NEXT N: RETURN
6600 LET F=INT (RND*52)+1
6610 IF A$(F,4)<>"0" THEN  GO TO
 1620
6620 LET X=0
6630 IF A$(F,1)="2" OR A$(F,1)="
4" THEN  LET X=2
6640 RETURN
7000 LET F=INT (RND*52)+1
7010 IF A$(F,1)<>"0" THEN  GO TO
 7000
7020 LET X=0: IF A$(F,4)="A" OR
A$(F,4)="D" THEN  LET X=2
7030 RETURN
7040 IF A$(F,2 TO 3)=" A" THEN
LET V=14: RETURN
7050 IF A$(F,2 TO 3)=" J" THEN
LET V=11: RETURN
7060 IF A$(F,2 TO 3)=" Q" THEN
LET V=12: RETURN
7070 IF A$(F,2 TO 3)=" K" THEN
LET V=13: RETURN
7080 LET V=VAL A$(F,2 TO 3)
7090 RETURN
8000 DIM A$(52,4): DIM P(5): DIM
 D(5)
8001 LET A$(1)="0 AA"
8002 LET A$(2)="0 2A"
8003 LET A$(3)="0 3A"
8004 LET A$(4)="0 4A"
8005 LET A$(5)="0 5A"
8006 LET A$(6)="0 6A"
8007 LET A$(7)="0 7A"
8008 LET A$(8)="0 8A"
8009 LET A$(9)="0 9A"
8010 LET A$(10)="010A"
8011 LET A$(11)="0 JA"
8012 LET A$(12)="0 QA"
8013 LET A$(13)="0 KA"
8014 LET A$(14)="0 AB"
8015 LET A$(15)="0 2B"
8016 LET A$(16)="0 3B"
8017 LET A$(17)="0 4B"
8018 LET A$(18)="0 5B"
8019 LET A$(19)="0 6B"
8020 LET A$(20)="0 7B"
8021 LET A$(21)="0 8B"
8022 LET A$(22)="0 9B"
8023 LET A$(23)="010B"
8024 LET A$(24)="0 JB"
8025 LET A$(25)="0 QB"
8026 LET A$(26)="0 KB"
8027 LET A$(27)="0 AC"
8028 LET A$(28)="0 2C"
8029 LET A$(29)="0 3C"
8030 LET A$(30)="0 4C"
8031 LET A$(31)="0 5C"
8032 LET A$(32)="0 6C"
8033 LET A$(33)="0 7C"
8034 LET A$(34)="0 8C"
8035 LET A$(35)="0 9C"
8036 LET A$(36)="010C"
8037 LET A$(37)="0 JC"
8038 LET A$(38)="0 QC"
8039 LET A$(39)="0 KC"
8040 LET A$(40)="0 AD"
8041 LET A$(41)="0 2D"
8042 LET A$(42)="0 3D"
8043 LET A$(43)="0 4D"
8044 LET A$(44)="0 5D"
8045 LET A$(45)="0 6D"
8046 LET A$(46)="0 7D"
8047 LET A$(47)="0 8D"
8048 LET A$(48)="0 9D"
8049 LET A$(49)="010D"
8050 LET A$(50)="0 JD"
8051 LET A$(51)="0 QD"
8052 LET A$(52)="0 KD"
8053 RETURN
9000 RESTORE : FOR a=USR "a" TO
USR "k"+7
9010 READ user: POKE a,user
9020 NEXT a: GO TO 10
9030 DATA 24,60,126,255,255,126,
60,24
9040 DATA 28,28,8,107,127,107,8,
28
9050 DATA 16,56,124,254,254,254,
16,56
9060 DATA 102,255,255,255,255,12
6,60,24
9070 DATA 204,204,51,51,204,204,
51,51
9080 DATA 0,0,0,7,15,12,24,24
9090 DATA 0,0,0,192,240,48,24,24
9100 DATA 24,24,12,15,7,0,0,0
9110 DATA 24,24,48,240,192,0,0,0
9120 DATA 0,0,0,255,255,0,0,0
9130 DATA 24,24,24,24,24,24,24,2
4
9990 REM a b c d e f g h i j k
9991 REM A B C D E F G H I J K
9992 BORDER 7: PAPER 7: INK 0: C
LS : STOP
9995 SAVE "STUD POKER" LINE 9000
9996 GO TO 10
```

WARREN'S 2068 BASIC'S

Horizontal Bar Chart

Warren Fricke

HORIZONTAL BAR CHART is a routine in BASIC for the TS2068 with an attached TS2040 printer. The purpose of the routine is two-fold. One, it is a relatively short program that provides a bar chart to illustrate the comparative status of a number of items (up to 20) in an inventory. Two, it demonstrates one way in which the 2040 printer can print out and "couple" automatically several screens full of data, and print it as a single illustration.

This program will print out one screen full, or two screens full, as the number of items requires.

For example, we input data on 15 assumed and related items, in this case various fruits. We also included make-believe quatities and a title, in response to LINE 55 of the program. The routine is universal. You may enter any related items and respective quantities. The item name is limited to 12 characters, in LINE 20. We also elected to call this FIGURE 1, but again, this may be changed to suit your application. Usually bar charts are used for a group of related items of 5 or more. Less than this and a pie chart may be a better choice.

In order to understand the arithmetic of the PLOT and PRINT AT lines, such as 240, 250, 630 and 640, you should refer to a screen chart for the computer. There is one on page 152 of the User Manual.

```
  1 REM ** HORIZONTAL BAR CHART
  2 REM ** "D-30", 3-17-88, WF

  5 REM ** ENTER DATA
  8 LET max=0: POKE 23658,8
 10 INPUT "How many items, 5 to
20? ";I
 15 IF I>20 THEN GO TO 10
 20 DIM A$(I,12): DIM Q(I)
 30 FOR n=1 TO I
 40 INPUT "Enter item No. ";(n)
;", and quantity ";A$(n),Q(n)
 45 IF Q(n)>max THEN LET max=Q(
n)
 50 NEXT n
 55 INPUT "Enter title.  Maximu
m of 32      characters. ";B$
200 REM ** PLOT DATA
210 LET V=160
220 FOR m=1 TO I
230 FOR n=0 TO 2
240 PLOT 0,V+8+n: DRAW 253*Q(m)
/max,0: NEXT n
250 PRINT AT 21-V/8,1;A$(m);" =
";Q(m);" units."
260 LET V=V-16
270 IF m=10 AND I=10 OR m>10 AN
D I>10 THEN GO TO 500
280 NEXT m
290 PRINT AT 19,16-LEN B$/2;B$;
AT 20,12;"FIGURE 1"
300 PLOT 0,0: DRAW 0,175: DRAW
255,0: DRAW 0,-175: DRAW -255,0:
 COPY : STOP
```

```
APPLES          = 100 units.
AVOCADOS        = 20 units.
BANANAS         = 2 units.
BLUE BERRIES    = 50 units.
CHERRIES        = 72 units.
GRAPES          = 15 units.
GRAPE FRUIT     = 25 units.
LEMONS          = 80 units.
LIMES           = 11 units.
ORANGES         = 115 units.
PEACHES         = 62 units.
PEARS           = 35 units.
PINEAPPLES      = 27 units.
PLUMS           = 12 units.
TANGERINES      = 7 units.

        INVENTORY OF FRUIT
           FIGURE 1
```

```
500 PLOT 0,0: DRAW 0,175: DRAW
255,0: DRAW 0,-175: COPY : CLS
510 IF I>=12 THEN GO SUB 600
520 PRINT AT 19,16-LEN B$/2;B$;
AT 20,12;"FIGURE 1"
530 PLOT 0,175: DRAW 0,-175: DR
AW 255,0: DRAW 0,175: COPY : STO
P
600 REM ** PLOT CONTINUATION
                of DATA
605 LET V=160
610 FOR m=12 TO I
620 FOR n=0 TO 2
630 PLOT 0,V+8+n: DRAW 253*Q(m)
/max,0: NEXT n
640 PRINT AT 21-V/8,1;A$(m);" =
";Q(m);" units."
650 LET V=V-16
670 NEXT m: RETURN
```

A BROKEN 2068? FIX IT YOURSELF!

by John M. Bell

IMPORTANT NOTE: The following article describes a method of testing for, and repairing a Timex Sinclair 2068 with bad RAM chips. Though it is a simple project that most people with basic soldering skills are capable of completing, and the information presented here is believed to be correct, the author and publisher take no responsibility for any damage done to the computer (or hobbyist) as a result of, or while using this information.

The most common cause of a Timex Sinclair 2068 not working is burnt-out RAM chips. The chip(s) are usually destroyed when turning on the computer with a printer or disk interface attached. Though I have no way of proving it, the most likely cause is a high voltage "glitch" produced by the switching regulator when it is "powered-up" under a heavy load. Note that one or all of the chips can burn out, and in any combination.

There are three symptoms that may indicate bad RAM chips. The first is a blank screen when the computer is turned on. This suggests either a "brain

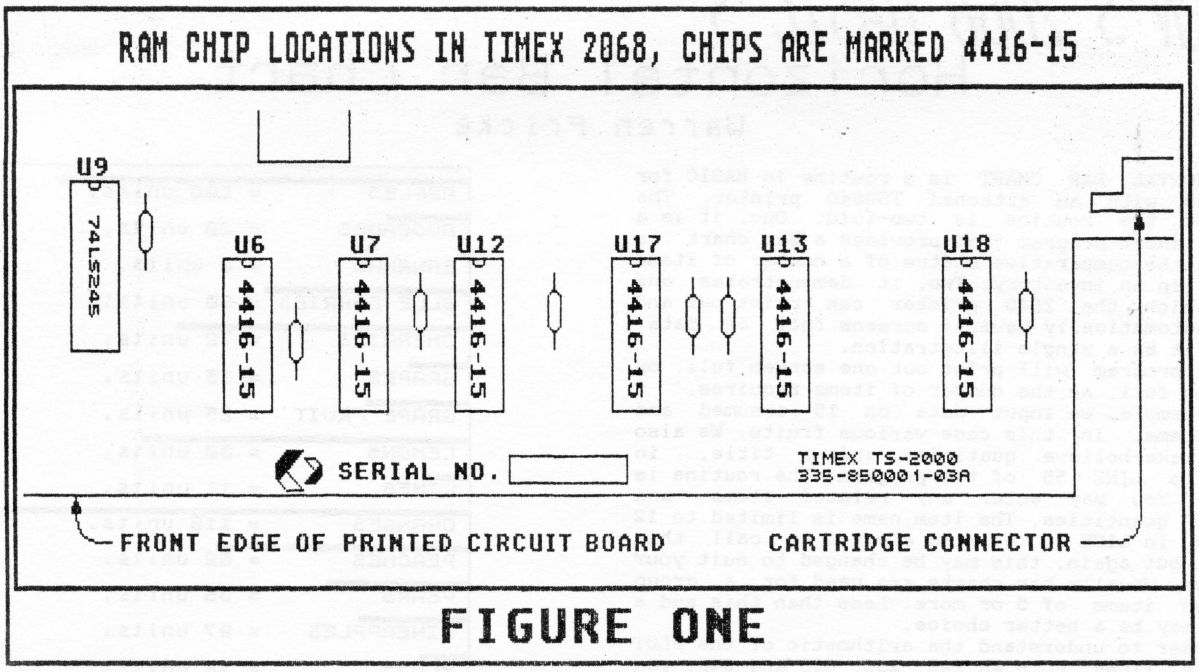


FIGURE ONE

dead" 2068 (the SCLD...or "square chip" is destroyed, and is VERY difficult to replace, IF you can get one), or that most of the RAM chips have burnt out. The second symptom is a display that consists of a white border around a screen of "garbage". This indicates that at least one of the RAM chips in the first 16K bank is bad. The last and most obvious symptom is the free memory after power up is less than 38652 bytes (using the PRINT FREE command). If a Spectrum ROM is installed in the computer, the original Timex ROM will have to be replaced to use the FREE command.

If your machine exhibits any of these symptoms, there is a chance it can be easily repaired. First, the computer's PC board must be completely removed from it's case. Save all the screws and be careful to avoid damaging the keyboard ribbon cable. Place the PC board on a non-conductive work surface and plug it in. Leave it turned on for a few minutes and then check each of the RAM chips for overheating (see Figure One for the chip locations). If any of the chips are hot to the touch, they are bad and need replacing. Mark them for removal. If none of the chips are hot and the computer still displays a black screen, the problem is probably not with the RAM chips. Consider sending the machine out for repair (to Dan Elliott of Promise Land Electronics--see May/June '88 issue of TDM for address listing). If the computer displays a border around a screen of garbage, chips U6 and/or U7 may have gone bad. If a normal sign on the screen is displayed, but only 22268 bytes are "free", chips U16 and/or U17 may have gone bad. If 5884 bytes are free, chips U12 and/or U13 and possibly U16 and/or U17 may have gone bad.

The chips that are hot to the touch should now be removed. Don't bother trying to remove them in one piece. Just cut or clip pins near the body of the chip, and remove the remaining pins from the PC board with a hot soldering iron and tweezers. If the chips are suspected to be bad but are not getting hot, a more difficult problem exists. The chips can be clipped off the board and discarded, or removed in one piece. If the chip is clipped off, you will never know if it was good or bad, and it will have to be replaced. If the chip is removed in one piece (a very difficult task), the chip can be saved for testing and possible re-use...but only at the risk of possibly damaging the PC board. Make your own decision.

Once the chips have been removed, the computer should be tested again. Connect it to a monitor and turn it on. If any of the remaining RAM chips are now getting hot, they should also be removed. If the display was formerly black, and now displays the normal copyright message, but with reduced RAM available, the chips can be replaced and computer should work fine. If a border is displayed around a screen of garbage, and chips U6 and U7 are still on the board, one or both of them may also be bad. Remove them for replacement or testing.

New RAM chips for the computer will have to be purchased. The 2068 uses 4416-15's, which are 16K*4 RAM chips. The 15 in the chip number designates speed. In this case 150 ns (nano seconds). Purchase either 120 or 150 ns chips, as the slower 200 ns chips will not work. Radio Shack does not sell them, so they must be mail-ordered (suppliers listed at the end of the article). Consider purchasing extra RAM chips, so that if one of the RAM chips left on the board is bad, you won't have to re-order. Don't even think about soldering the chips in. Purchase IC sockets along with the chips. Sockets make it easy to remove a chip for testing, and at twenty cents each, are a lot cheaper than cutting a $4.00 RAM chip off the board.

The sockets should now be soldered in place where the RAM chips once were. If the holes on the PC board are filled with solder, they will first have to be cleared. The best method I have found is to hold the PC board vertically in a vice, melt the solder from one side of the board with a soldering iron and use a solder pump to suck the molten solder out from the other side. Remove any solder splashes or excess flux from the PC board and solder the sockets into place using rosin core solder.

Insert the new RAM chips into the sockets with the notch end of the chip pointing to the back of the PC board. Test the computr out of the case once again as described in paragraphs two and three. If everything checks out fine, the computer can be re-installed in the case and used as normal. If the computer still does not work or has reduced RAM available, there are two possible reasons. Either there is yet another bad RAM chip (new or old), or another chip in the computer is damaged, but still operates. Re-test the computer for bad RAM chips, and if none can be found, consider having the computer professionally repaired.

SUPPLIERS:

JDR Microdevices, 110 Knowles Drive, Los Gatos, CA 95030, (800) 538-5000. Takes VISA & M-CARD, $10 min. order.

JAMECO Electronics, 1355 Shoreway Road, Belmont, CA 94002, (415) 592-8097. Takes VISA & M-CARD. $20 min. order.

curry computer

P.O. BOX 5607
GLENDALE, AZ 86312-5607
(602) 978-2902

***** SUMMER CLEARANCE SALE *****

SOFTWARE for the T/S 1000: All $2 ea.
Mixed Game Bag I\Presidents\ Stk Mkt Calc\ Red Alert\ Night Gunner\ Hangman\ Fin Mgr & Rec Keep\ Alien Invasion\ Meteorites\ Chess\ Gambler\ Cube Game\ Mixed Game II\ Organizer\ Home Asset\ Home Improve.\ VuCalc\ Geometry\ Pioneer Trail\ Damper-Glooper\ Croaka Crawler Plus many more - Write for a list.

T/S 1000 Computer (2K) with 3 Programs - $22.95

SOFTWARE for the T/S 2068:
Budgeter...$5 Stk Mkt Calc...$3.50
Fighter Pilot...$12.95

SOFTWARE for the Spectrum:
Storm Bringer...$5 I, Ball...$6 Biggles...$8
10th Frame...$10 Mag Max...$10 Snowman...$5
Speed King 2...$5 Skyranger...$6 Plus many 'One Onlys', write for a list.

SOFTWARE for the QL:
Super Disk (TKII req.)...$19.95 Cribbage...$14.95
SuperBoot...$14.95 (disk) QRAM...$39.95
Graphic TK...$18.95 Wanderer (RGB only)...$18.95
Archive 2.38...$14.95 Grab Bag II...$14.95
Presidents...$9.95 Nucleon...$19.95
Super Checking...$14.95 Thompson Case (12)...$1.99
Assembler...$34.95

MAGAZINES:
QL World: Current Issues - $4.25 \ Back Issues $3 (Jan/Feb/Mar/Ap/May/June/Oct '87)
Sinclair User: Current Issues - $4.50 \ Back Issues $2 (call for list)
We also have: Commodore User; Commodore International; Amiga User; Atari User; Atari ST User; Computer & Video Games; PC Amstrad; Amstrad User; Amstrad PCW...Call for Pricing.

S/H Charges: Under $15 = $1 \ Under $30 = $2 \ Under $50 = $3 \ Under $100 = $4 \ Under $200 = $6 \ Over $200 = $8

Sale Prices Good for 30 Days from Publication.

Mass-Storage

CASSETTE TAPE STORAGE

2068 PROGRAM INDEX

by V. Phillip Hosey

This simple program is easily adaptable to virtually any computer. It not only provides a modicum of security, but files all your stored programs by digital location on tape, eliminating a lengthy title-by-title search. I always incorporate an attention-getting BEEP at the beginning of each program SAVEd LINE 1 so that I may divert to other productive activities while passing the few minutes delay required for LOADing lengthy data. One last note: LINE 0 was obtained from Randall Larson's "NO DELETE PROGRAM" listed in the NOV/DEC '87 issue of TDM ("In The Mailbag"). Originally intended for the Spectrum, it functions perfectly without modification on the TS2068.

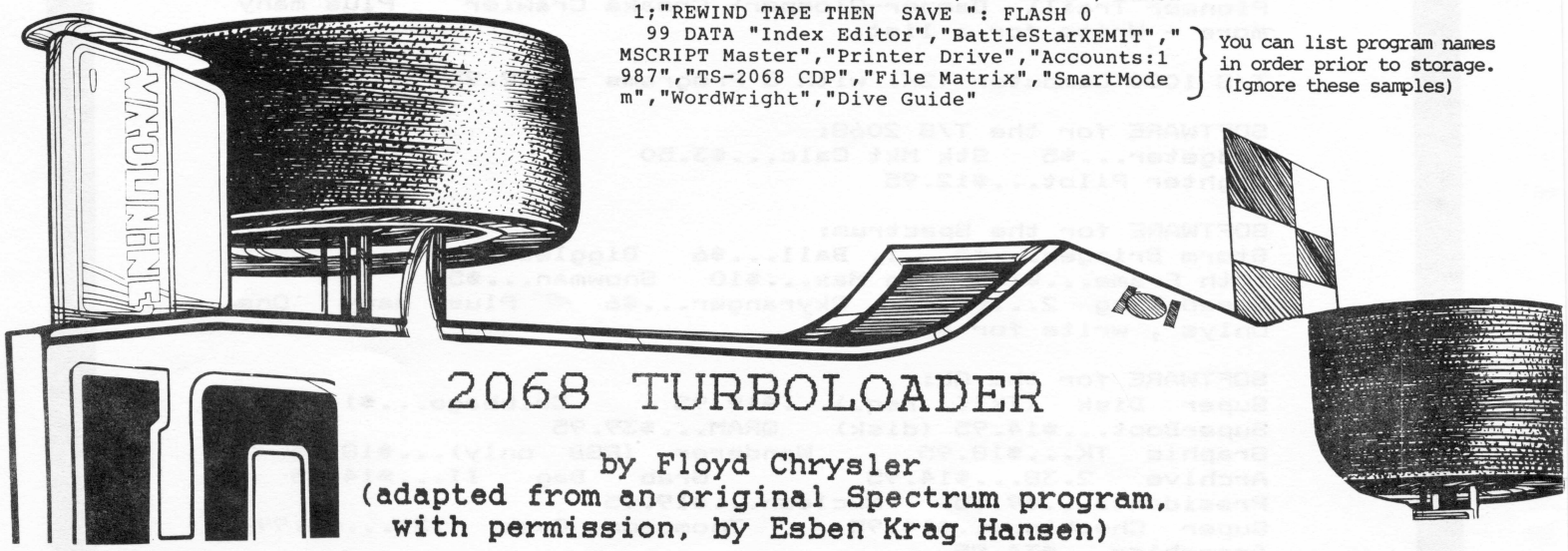
```
  0>REM 1985:V. Phillip Hosey
  1 PRINT "STOP TAPE:CODE?": INPUT A$: IF A
$<>"(any CODE you want)"  THEN  NEW
 10 BORDER 0: PAPER 0: FOR 1=0 TO 21: CLS :
 PRINT  INK 7;AT 1,0;": V. Phillip Hosey/T
S-2068:1985": BEEP .01,-1: NEXT 1
 20 INK 2: CIRCLE 80,92,81
 23 INK 4: CIRCLE 207,55,43
 25 INK 1: CIRCLE 198,136,32
 30 PRINT  INK 6;AT 2,10;"1";AT 3,7;"Progra
m";AT 4,7;"-INDEX-";AT 14,23;"16k=90";AT 15
,23;"second";AT 16,22;"loadtime";AT 4,22;"I
NKEY$"; FLASH 1;AT 5,24;"#?": FLASH 0
 40 FOR n=0 TO 9: READ A$: PRINT  INK 3;AT
n+6,2;n;" "; INK 7;A$: NEXT n
 50 FOR y=0 TO 31: PRINT  INK 7;AT 21,y;"."
;AT 21,y;" "
 60 IF INKEY$="" THEN  NEXT y: BEEP .01,9:
GO TO 50
```

I use my computer's serial number as an access code.

```
 65 IF INKEY$="0" THEN  GO TO 95
 81 IF INKEY$="1" THEN  LET t=15
 82 IF INKEY$="2" THEN  LET t=46
 83 IF INKEY$="3" THEN  LET t=56
 84 IF INKEY$="4" THEN  LET t=70
 85 IF INKEY$="5" THEN  LET t=110
 86 IF INKEY$="6" THEN  LET t=145
 87 IF INKEY$="7" THEN  LET t=165
 88 IF INKEY$="8" THEN  LET t=200
 89 IF INKEY$="9" THEN  LET t=300
 90 PRINT  INK 5;AT 3,22;"Start ";AT 4,22;"
 Tape ";AT 5,22;"  Now ";AT 14,23;"Fast- ";
AT 15,23;"Forward";AT 16,22;" Tape To"; FLA
SH 1;AT 17,25;t;AT 21,7;"* PAUSE  TO  LOAD
";INKEY$; FLASH 0: INK 7: LOAD INKEY$
 95 CLS : LIST 99: BEEP 1,30: PRINT ; FLASH
 1;"REWIND TAPE THEN  SAVE ": FLASH 0
 99 DATA "Index Editor","BattleStarXEMIT","
MSCRIPT Master","Printer Drive","Accounts:1
987","TS-2068 CDP","File Matrix","SmartMode
m","WordWright","Dive Guide"
```

You will insert these digital locations after SAVEing your programs. (Ignore these samples)

You can list program names in order prior to storage. (Ignore these samples)



2068 TURBOLOADER

by Floyd Chrysler
(adapted from an original Spectrum program, with permission, by Esben Krag Hansen)

Do you have lots of programs on tape? Do you go crazy waiting for them to load? If you answered yes then you may be interested in this program to double the speed of your tape loads.

I was in the same position when I received the May 1986 issue of "Your Sinclair". Even though I have the AERCO FD-68 I still have lots of programs on tape I really didn't want to move to disk and I still make backup copies of important progams for long term storage. Using disk made my tape deck seem so slow.

In the magazine was a program by Esben Hansen for the Spectrum that allowed you to save and load programs at variable baud rates from 1500 (normal rate) to 3500 (more than double). I wanted to do this on a 2068 so I vowed to convert the program to work on my machine.

This was not as easy as I had thought. While a lot of the program was compatable with the 2068, by just changing the ROM calls, the calls to the tape handling routines were a problem. On the 2068 the tape routines are all in EXROM. It was either bankswitching or a rewrite to do the tape handling routines within the program. I chose the second and by some fiddling managed to get the program to work with most of the functions in the original and only use a little over 200 bytes more.

The program supports all the tape commands - Save, Load, Verify, and Merge. Turboloader is invoked by a RANDOMIZE USR 63600. This is followed by the Turboloader commands - LIST, PRINT, INPUT, or RUN. And finally your tape command - SAVE, LOAD, etc.

To save a screen at double speed:

```
RANDOMIZE USR 63600: RUN 3100: SAVE "pic" SCREEN$
```

The keywords List, Print, Input, and Run will still work as normal except when they follow a RANDOMIZE USR 63600. When used as Turboloader commands they have the following meaning:

LIST - This command reads a header from tape and lists it on the screen. It displays program length, data length for code, start line, etc.
ex: RANDOMIZE USR 63600: RUN 3100: SAVE "test" CODE 63600,600: LIST

RUN - This changes the baud rate. It must be followed by a number ranging from 1500 to 3500, in steps of 200 (1500,1700,1900, etc.). A good tape recorder should be able to handle at least 3100.

INPUT- This deals with the message start tape on SAVE's
INPUT 2 prints the Turboloader message and the start tape message and waits for a keypress.
INPUT 1 print the Turboloader message only and waits for a key press.
INPUT 0 prints nothing and goes right into the save without waiting for a key press.

PRINT- This deals with LOAD/VERIFY commands
Print 2 prints the turboloader message and program names as they are found.
Print 0 nothing is printed.

PRINT 2 and INPUT 2 are the initial default values.

There is a lot of code to enter for Turboloader. I have included a Hex loader program to make it a little easier. Enter the loader program and save it. The code is listed in three columns. The first is the address for that code line. Next is the code in blocks of 8 hex bytes. Last is a check digit.

When you run the hex loader it will ask you for a start address. The first time you should enter 63380. It will then ask for the code. Enter all 16 characters (8 hex bytes) and press enter. You will then be asked for the check number. Enter it and if all is ok the program will display the next address (which should match the next address in the code list), if there is an error the program will beep and redisplay the same address for you to reenter the line in error.

At any time you may enter STOP to the enter code message and you will be prompted to save the code entered to that point. You can then restart at a later time by reloading the code and loader program and entering the next address from where you left off at the prompt. When you have entered all the code the program will prompt you to save the code. Once you have all the code saved you can start speeding! Enter CLEAR 63379: LOAD "TURBO" CODE 63380. Remember the entry point to the program for all user calls is 63600.

Don't be intimidated by all the code. I'm sure you will find it worth the time to enter. If you have not yet spent the family fortune on a disk drive you will find this program invaluable.

Hex Loader

```
 30 DEF FN h(a$)=CODE a$-48-(7
AND a$>"9")
 40 POKE 23658,8
 50 INPUT "Start Address:";star
t
 60 IF start=63380 THEN  GO TO
90
 70 INPUT "Have you re-loaded c
ode?";z$
 80 IF z$<> "Y" THEN  PRINT "Lo
ad code and re-start": STOP
 90 LET adr=start
 100 FOR t=start TO 65444 STEP 8

 110 LET sum=0
 120 PRINT adr;" ";
 130 INPUT "enter code ";c$
 140 IF c$="STOP" THEN  GO TO 38
0
 150 IF LEN c$<>16 THEN  BEEP .2
5,.25: GO TO 130
 160 FOR i=1 TO 8
 170 LET a$=c$(2)
 180 POKE adr,FN h(c$)*16+FN h(a
$)
 190 PRINT c$(1);a$;
 200 LET c$=c$(3 TO )
 210 LET sum=sum+(PEEK adr)
 220 LET adr=adr+1
 230 NEXT i
 240 INPUT "Check Number=";check

 250 PRINT " ";sum
 260 IF sum<>check THEN  GO TO 3
40
 270 NEXT t
 280 PRINT "END OF CODE"
 290 SAVE "TURBO"CODE 63380,2066

 300 PRINT "VERIFY"
 310 VERIFY "TURBO"CODE 63380
 320 PRINT "FINISHED"
 330 STOP
 340 LET adr=adr-8
 350 BEEP .5,1
 360 PRINT "ERROR - RE-ENTER"
 370 GO TO 110
 380 INPUT "Do You want to Save
your work?  (Y or N)";z$
 390 IF z$="Y" THEN  GO TO 290
 400 STOP
 999 SAVE "hexload" LINE 10
```

Turboloader Code (ready for Hex Loader program)

addr	code	check
63380	CF1A7EE6C020191A	864
63388	13BE2320021ABE1B	521
63396	2B3008E5EBCD2017	823
63404	E118ECCDF0F718E2	1427
63412	7E4FFE80CBE5214B	1124
63420	5C7EFE802825B928	902
63428	08C5CD2017C1EB18	917
63436	F0E6E0FEA02012D1	1367
63444	D5E523131ABE2006	750
63452	1730F7E11803E118	819
63460	E03EFFD1EB3C37CD	1305
63468	F0F718C420100822	797
63476	5F5CEBCD2017CD50	967
63484	17EB2A5F5C0808D5	716
63492	CD2017225F5C2A53	606
63500	5CE3C50838072BCD	835
63508	BB12231803CDBB12	677
63516	23C1D1ED53535CED	1169
63524	5B5F5CC5D5EBEDB0	1336
63532	E1C1D5CD5017D1C9	1349
63540	3D20FDA704C83E7F	906
63548	DBFE1FD0A9E62028	1183
63556	F3792F4FE607F608	981
63564	D3FE37C9F53A485C	1188
63572	E6380F0F0FD3FE3E	858
63580	7FDBFE1FFB3802CF	1147
63588	0CF1C900000000000	454
63596	00000000DFFE0D28	530
63604	47FE3A2802CF0BFD	896
63612	340D2A5D5CE5E721	785
63620	70F8E3FEF5285DF5	1464
63628	E5CDCDF8E1F1FEEE	1845
63636	285FFEF7CA27FAFE	1381
63644	F0CA7BFA0600FEF8	1320
63652	286104FEEF285C04	770
63660	FED6285704FED528	1106
63668	52225D5CFD350DE1	845
63676	CDC4F8ED4B765CC9	1372
63684	3A3BFFFE02C0C3A9	1184
63692	083A3CFFFE02C0CD	1034
63700	A9081160FEAFCD3F	987
63708	07ED4B37FFCD8817	993
63716	3E0DD7C9CD00F9FE	1199
63724	03D276FA323BFF18	969
63732	D8CD00F9FE0330F1	1216
63740	323CFFC9E7CDE51B	1258
63748	C31E1F783239FFE7	969
63756	CDEF1BCD8928283D	954
63764	0111003A39FFA728	595
63772	020E22F7D5DDE106	962
63780	0B3E20121310FCDD	631
63788	3601FFCDAF2F21F6	1016
63796	FF0B090330103A39	457
63804	FFA72003C3701478	904
63812	B1280A010A00DDE5	688
63820	E123EBEDB0DFFEE4	1613
63828	20473A39FFFE03CA	932
63836	ED1BE7CD702CCBF9	1308
63844	300C2100003A39FF	463
63852	3D2816C3911BC2ED	921
63860	1BCD89282818237E	634
63868	DD770B237EDD770C	864
63876	23DD710E3E01CB71	762
63884	28013CDD7700EBE7	907
63892	FE2920DAE7EBC36D	1315
63900	FBFEAA201C3A39FF	1105
63908	FE03CAED1BE7DD36	1229
63916	0B00DD360C1B2100	358
63924	40DD750DDD740E18	790
63932	4AFEAF204D3A39FF	982
63940	FE03CAED1BE7CDE7	1390
63948	21200C3A39FFA7CA	816
63956	ED1BCD511C180FCD	822
63964	E51BDFFE2C280C3A	887
63972	39FFA7CAED1BCD51	1231
63980	1C1804E7CDE51BCD	953
63988	231FDD710BDD700C	756
63996	CD231FDD710DDD70	951
64004	0E6069DD360003C3	688
64012	6DFBFECA2807DD36	1138
64020	0E80C34DFB3A39FF	1035
64028	A7C2ED1BE7CDE51B	1317
64036	C344FBE7CDE51BCD	1411
64044	231FC5CDE9300164	850
64052	00CDE930EF0538CD	991
64060	1E1FFE0F3834FE24	728
64068	3030D60FCB47202A	673
64076	CB27CB27214DFF06	855
64084	004F09EB213DFF06	678
64092	08C51A134E234623	468
64100	02C110F53A1BFE32	845
64108	49FEC1ED4337FFC3	1329
64116	CDF8CF09E7FE0D28	1207
64124	06FE3A2802CF0B11	595
64132	1100DD2126FFAF37	794
64140	CDCBFD30F23A26FF	1302
64148	FE0430EBCDC4F83E	1252
64156	02CD301211F6FECD	995
64164	3F071127FF060A1A	423
64172	FE203001AFD71310	760
64180	F63E0DD73A26FFA7	1054
64188	2822FE03285D3E05	531
64196	1160FECD3F073A34	752
64204	FFF5E61FC640D7F1	1479
64212	CB7728033E24D73E	740
64220	0DD7184EED4B33FF	948
64228	78E6C02010C53E02	851
64236	1160FECD3F07C1CD	1040
64244	88173E0DD73E0611	534
64252	60FECD3F07ED4B35	990
64260	FFC5CD3AFB3E0711	1052
64268	60FECD3F07C12A31	909
64276	FFA7ED42444DCD3A	1133
64284	FB180F3E031160FE	722
64292	CD3F07ED4B33FFCD	1098
64300	3AFB3E041160FECD	947
64308	3F07ED4B31FFCDE9	1124
64316	30CDA1313E0DD7C9	954
64324	CD231FDD710DDD70	951
64332	0EDD3600002A595C	512
64340	ED5B535C37ED52DD	1098
64348	750BDD740C2A4B5C	686
64356	ED52DD750FDD7410	1025
64364	EB3A39FFA7CA0DFD	1240
64372	E5011100DD09CDCD	887
64380	F8DDE5111100AF37	962
64388	CDCBFDDDE130EF3A	1452
64396	3BFFA7280CCDC4F8	1182
64404	3EFECD3012FD3652	976
64412	030E80DD7E00DDBE	903
64420	EF20020EF6FE0430	839
64428	CD473A3BFFA77828	975
64436	0B11F6FEC5CD3F07	997
64444	C1DDE5D121F0FF19	1405
64452	060A7E3C20037980	486
64460	4F131ABE2320010C	394
64468	3A3BFFA728021AD7	822
64476	10EFCB7920983A3B	880
64484	FFA728063E0DD7CD	963
64492	CDF8E1DD7E00FE03	1282
64500	280C3A39FF3DCA41	750
64508	FCFE02CAEEFCE5DD	1650
64516	6EFADD66FBDD5E0B	1260
64524	DD560C7CB5280EED	915
64532	523827280BDD7E00	572
64540	FE03C294F7E17CB5	1376
64548	2006DD6E0DDD660E	719
64556	E5DDE13A39FFFE02	1301
64564	372001A73EFFCDCB	980
64572	FDD8C394F7DD5E0B	1385
64580	DD560CE57CB52006	891
64588	131313EB180CDD6E	659
64596	FADD66FBEB37ED52	1433
64604	3B0911050019444D	257
64612	CDBB1FE1DD7E00A7	1162
64620	283D7CB528132B46	578
64628	2B4E2B030303DD22	428
64636	5F5CCD5017DD2A5F	853
64644	5C2A595C2BDD4E0B	668
64652	DD460CC5030303DD	730
64660	7EFDF5CDBB1223F1	1310
64668	77D12373237223E5	891
64676	DDE1373EFF188FEB	1220
64684	2A595C2BDD225F5C	708
64692	DD4E0BDD460CC5CD	1015
64700	4D17C1E5C5CDBB12	1129
64708	DD2A5F5C23DD4E0F	799
64716	DD461009224B5CDD	738
64724	660E7CE6C0200ADD	925
64732	6E0D22425CFD360A	632
64740	00D1DDE1373EFFC3	1222

```
addr  code                check
64748 3AFCDD4E0BDD460C  923
64756 C503F73680EBD1E5  1302
64764 E5DDE1373EFFCD3A  1310
64772 FCE1ED5B535CC396  1325
64780 F7E53A3CFFA72824  1092
64788 3D200A3EFDCD3012  689
64796 CDD9111817CDA908  868
64804 CDCDF83E011160FE  1088
64812 CD3F07CDD911CDA9  1088
64820 08CDCDF8DDE51111  1150
64828 00AFCD52FDDDE106  1167
64836 32761OFDDD5E0BDD  984
64844 560C3EFFDDE12150  974
64852 F8E521801FCB7F28  1039
64860 0321980C0813DD2B  491
64868 F33E04471OFED3FE  1115
64876 EE0F06A42D20F505  750
64884 25F268FD062F10FE  959
64892 D3FE3E0C063710FE  870
64900 D3FE010D3B086F18  681
64908 097AB3280BDD6E00  692
64916 7CAD673E02371817  566
64924 6C18F579CB7810FE  1091
64932 3004064210FED3FE  859
64940 063E20EF05AF3CCB  782
64948 1520EB1BDD230631  626
64956 3E7FDBFE1FD07A3C  1083
64964 20C7063B10FEC914  787
64972 0B15F33E0FD3FE21  847
64980 50F8E5DBFE1FE620  1323
64988 F6044FBFC0CD5BFE  1262
64996 30FA2115041OFE2B  669
65004 7CB520F9CD57FE30  1180
65012 EB069CCD57FE30E4  1219
65020 3EC6B830E02420F1  1025
65028 06C9CD5BFE30D578  1138
65036 FED430F4CD5BFED0  1516
65044 79EE064F260006B0  664
65052 181F082007300FDD  386
65060 75001B0FCB11ADC0  741
65068 791F4F131807DD7E  628
65076 00ADC0DD231B0B06  662
65084 B22E01CD57FED03E  1041
65092 CBB8CB1506B030F3  1084
65100 7CAD677AB320CB7C  1060
65108 FE01C9CD5BFED03E  1276
65116 16C334F880545552  896
65124 424F4C4F4144207F  592
65132 313938362045482B  432
65140 4643202042415544  485
65148 BA2020535441525 4  648
65156 2054415045204 14E  505
65164 442050524553532 0  529
65172 414E592042B45D953  708
65180 5441525420 4C494E  574
65188 45BA535441525420  685
65196 4144445245535 3BA  704
65204 4C454E47544B204F  561
65212 462044415441204 2  482
65220 4C4F434BBA415252  712
65228 4159204E414D45BA  661
65236 50524F4752414D20  568
65244 4C454E475448BA4C  712
65252 454E475448204F46  555
65260 2056415249414 24C  545
65268 4553BA0D50524F47  663
65276 52414DBA0D4E554D  663
65284 4245522041525241  543
65292 59BA0D4348415241  639
65300 435445520415252  563
65308 4159BA0D42595445  661
65316 53BA0000000000000  269
65324 0000000000000000  0
65332 000000DC05000002  227
65340 028BFDA7FDADFDBB  1424
65348 FD1BFE3CFE44FE5C  1262
65356 FE3B423E31B0B2CB  1047
65364 16373D3A2DB7B9D0  817
65372 153239352BBDBFD4  813
65380 132E343124C4C6D9  813
65388 1229302C1FCACCDD  809
65396 10252B281BD1D3E2  809
65404 0F21272417D7D9E6  808
65412 0D1C221F12DEE0EB  805
65420 0C1B1E1B0EE4E6EF  804
65428 0A13191609EBEDF4  801
65436 090F151205F1F3F8  800
65444 0700000000000000  7
```

BANK-SWITCHING RAM

PRACTICAL 2068 BANK-SWITCHING

by Stan Lemke

Although some bank-switching applications might encompass seemingly insurmountable programming obstacles...there are many others that can be accomplished with ease! I'd like to present three bank-switching examples/applications that can be adapted and expanded on for a wide variety of uses.

BANK SWITCHING THEORY--FROM A LAYMAN'S PERSPECTIVE

What is bank switching? In very simple terms, it is a way to direct the computer to switch between different "banks" of memory circuits. This is accomplished in a program with the OUT 244,VALUE command, where VALUE defines which memory "banks" are being used. Although the computer can only talk to 64K of memory (8 banks of 8K each) at any given time, BANK-SWITCHING can switch in/out different banks of memory...and make it appear like more memory. One little detail that makes this all possible, is that when you swtich out one memory bank for another, the memory in that bank remains just the way you left it...so when you return to it, you can continue on just as before!

Another important detail is that we will only be working with memory above location 32768. By doing this, we will not interfere with the computer operating system and greatly simplify our work. This means that we will only have an additional 32K of RAM to work with, but that almost doubles the memory capacity we are currently working with after subtracting that used by the operating system!

Memory is "bank-switched" in 8K chunks using the OUT 244,VALUE command. VALUE determines which chunks are being used. The following table defines VALUE and the "DOCK" memory addresses that are used.

Now, by adding various values, one can activate multiple banks of dock memory (i.e., VALUE = 64+128 = 192 operates on addresses 49152 - 65535). To reset all banks to the standard memory, use VALUE = 0. NOTE: We will only be activating chunks with addresses above 32768 (VALUE = 16 and above). AERCO FD-68 Users: The AERCO disk system requires that chunk 1 be active to utilize the disk, therefore, add 1 to your VALUE to keep the disk active.

Value	Dock Memory Addressed
1	0 - 8191
2	8192 - 16383
4	16384 - 24575
8	24576 - 32767
16	32768 - 40959
32	40960 - 49151
64	49152 - 57343
128	57344 - 65535

BANK-SWITCHING RAM (where to get it)

Add-on bank switchable RAM can be obtained from a variety of sources. The AERCO disk interface comes with 64K of additional bank-switchable RAM built right in. Another source is RAM cartridges that plug into the 2068 cartridge dock such as the one designed by Tom Bent (Quantum Levels), and once marketed by Thomas B. Woods, or the one available from Lem Software (see the ad on the back cover of this magazine for a 32K RAM cartridge). Other sources of RAM are available, like the new RAMdisk from LARKEN, and there are probably others that I am not aware of.

WHAT CAN WE DO WITH IT?

OK, what can we do with this add-on memory? What is the #1 complaint about 64K computers? They have so little memory to work with! There is always more data than memory to hold it! The #1 use for more memory will be to store more data. So, my first example is a short data transfer program.

XFER_1 (LISTING A)

XFER_1 is a ZEUS assembler source listing, ready to be assembled. LINE 270 is set to assemble this routine starting at 39000 (RANDOMIZE USR 39000). Following the source file is a dis-assembly of the

routine identifying the memory address, the value at that address, and the assembler instruction associated with that address. I would like to thank AL Schremmer, an active member of the Kansas Area TS User Group for writing this very helpful and unique dis-assembler!

XFER_1 is a simple program that will transfer (COPY) data from one bank to another bank. As written, 24064 bytes of data are copied from standard memory starting at address 41300 to the dock bank, also starting at address 41300. The "source" bank is the source of the data to be copied...and could be the dock bank. The "destination" bank is where you are copying data to, and could be regular memory (with a little modification). Also, this example copies data to the same address in the dock bank, but you can see that this also can be changed easily.

The way it works is this: after assigning the destination and source addresses, and the number of bytes to be copied (LINES 290 to 310), the source bank is activated (LINES 430 and 440) and 1 byte is copied into the accumulator and saved by pushing it onto the stack (LINES 450 and 460). Then the destination bank is activated (LINES 570 and 580) and the saved byte is recalled and stored at it's destination address (LINES 590 and 600). The destination and source addresses are incremented (LINES 670 and 680) while the number of bytes that are to be copied is decremented (LINE 690). The number of bytes remaining is checked to see if it is zero (LINES 700 to 720), and if not, the process is repeated. When complete, the source bank is activated (LINES 730 and 740), and the program ends with a RETURN.

XFER_2 (LISTING B)

A second form of data transfer useful in bank-switching is the ability to "swap" data between banks. This is the function of routine XFER_2. Where XFER_1 merely copied data from one bank/address to another, XFER_2 performs a swap function, copying the data from the source into the destination, and then placing the destination byte back into the source! This routine works much like XFER_1 above. The key to this program is the use of the AF and alternate AF registers, and exchanging these to easily allow the exchange of the source/destination values.

XFER_3 (LISTING C)

The third example is a merging of the source/destination data. This application superimposes the source data on the destination data using the "OR" function, and is equivalent of overlaying two pictures on a light board...ending up with one. I've used this function with my PIXEL PRINT PROFESSIONAL (desktop publisher) program, to combine (or merge) two PIXEL PRINT files. As you can see, the operation of the program is quite similar to the two above with a simple modification for the "OR" function. NOTE: both the source and destination addresses contain the merged data.

Now, I don't pretend to be a very good assembly programmer, so I am sure there are many other ways to do these jobs. But, if I have been able to show you enough to get your interest peaked, and convinced you that BANK-SWITCHING is not an impossible task, then I've accomplished my goal!

Keep TS2068-ing, and start taking advantage of 2068 BANK-SWITCHING.

XFER_1

```
39000 > 33    > ld hl,NN
39001 > 84      <41300>
39002 > 161
39003 > 17    > ld de,NN
39004 > 84      <41300>
39005 > 161
39006 > 1     > ld bc,NN
39007 > 0       <24064>
39008 > 94
39009 > 62    > ld a,N
39010 > 1       <1>
39011 > 211   > out (N),a
39012 > 244     <244>
39013 > 26    > ld a,(de)
39014 > 245   > push af
39015 > 62    > ld a,N
39016 > 225     <225>
39017 > 211   > out (N),a
39018 > 244     <244>
39019 > 241   > pop af
39020 > 119   > ld (hl),a
39021 > 19    > inc de
39022 > 35    > inc hl
39023 > 11    > dec bc
39024 > 120   > ld a,b
39025 > 177   > or c
39026 > 32    > jr nz,DIS
39027 > 237     <237>
39028 > 62    > ld a,N
39029 > 1       <1>
39030 > 211   > out (N),a
39031 > 244     <244>
39032 > 201   > ret
39033 > 0     > nop
```

LISTING A

```
00010 ;         XFER_1
00020 ; ----------------------
00030 ;       PRACTICAL
00040 ;     BANK-SWITCHING
00050 ;
00060 ;  ++++++++++++++++++++++
00070 ;  + THIS SAMPLE PROG. +
00080 ;  + WILL COPY 24064   +
00090 ;  + BYTES FROM MEMORY +
00100 ;  + ADDRESS 41300 OF  +
00110 ;  + STD. RAM TO 41300 +
00120 ;  + OF THE DOCK BANK. +
00130 ;  ++++++++++++++++++++++
00140 ;
00150 ; (c) S D LEMKE      1988
00160 ;
00170 ; LEMKE SOFTWARE DEVELOP.
00180 ;      2144 WHITE OAK
00190 ;   WICHITA, KS.  67207
00200 ; ----------------------
00210 ; ----------------------
00220 ;
00230 ; GETTING STARTED...
00240 ;  SET SOURCE, DEST, AND
00250 ;  NUMBER OF BYTES
00260 ;
00270 ORG 39000    ; CODE ADD.
00280 ;
00290 LD HL,41300 ; DEST. ADD.
00300 LD DE,41300 ; SOURCE ADD.
00310 LD BC,24064 ; LENGTH
00320 ;
00330 ;-------------------------
00340 ;
00350 ; BEGIN BY ENABLING THE
00360 ;  SOURCE BANK AND SAVING
00370 ;  THE SOURCE BYTE
00380 ;
00390 ; SOURCE BANK = 1 (AERCO)
00400 ; SOURCE BANK = 0 (OTHER)
00410 ;
00420 ;
00430 XFER1 LD A,1 ; SOURCE BNK
00440 OUT (244),A  ; ENABLE IT
00450 LD A,(DE) ; LOAD S. VALUE
00460 PUSH AF    ; SAVE IT
00470 ;
00480 ;-------------------------
00490 ;
00500 ; NEXT, ENABLE THE DEST.
00510 ;  BANK, AND STORE THE
00520 ;  SOURCE BYTE THERE
00530 ;
00540 ; DEST BANK = 225 (AERCO)
00550 ; DEST BANK = 224 (OTHER)
00560 ;
00570 LD A,225 ; DESTINATION
00580 OUT (244),A ; ENABLE IT
00590 POP AF ; RECALL S. VALUE
00600 LD (HL),A ; STORE IT
00610 ;-------------------------
00620 ;
00630 ; INCREMENT THE SOURCE
00640 ;  AND DEST. ADDRESSES.
00650 ;  CHECK TO SEE IF DONE.
00660 ;
00670 INC DE ; SOURCE + 1
00680 INC HL ; DEST. + 1
00690 DEC BC ; LENGTH - 1
00700 LD A,B ; "B" INTO ACC.
00710 OR C   ; SUM WITH "C"
00720 JR NZ,XFER1 ; "BC" = 0 ?
00730 LD A,1 ; RESTORE S. BANK
00740 OUT (244),A  ; ENABLE IT
00750 RET          ; ALL DONE
```

LISTING B

```
00010 ;         XFER_2
00020 ; ----------------------
00030 ;       PRACTICAL
00040 ;     BANK-SWITCHING
00050 ;
00060 ;  ++++++++++++++++++++++
00070 ;  + THIS SAMPLE PROG. +
00080 ;  + WILL SWAP 24064   +
00090 ;  + BYTES FROM MEMORY +
00100 ;  + ADDRESS 41300 OF  +
00110 ;  + STD RAM AND 41300 +
00120 ;  + OF THE DOCK BANK. +
00130 ;  ++++++++++++++++++++++
00140 ;
00150 ; (c) S D LEMKE      1988
00160 ;
00170 ; LEMKE SOFTWARE DEVELOP.
00180 ;      2144 WHITE OAK
00190 ;   WICHITA, KS.  67207
00200 ; ----------------------
00210 ; ----------------------
00220 ;
00230 ; GETTING STARTED...
00240 ;  SET SOURCE, DEST, AND
00250 ;  NUMBER OF BYTES
00260 ;
00270 ORG 39000    ; CODE ADD.
00280 ;
00290 LD HL,41300 ; DEST. ADD.
00300 LD DE,41300 ; SOURCE ADD.
00310 LD BC,24064 ; LENGTH
00320 ;
00330 ;-------------------------
00340 ;
00350 ; BEGIN BY ENABLING THE
00360 ;  SOURCE BANK AND SAVING
00370 ;  THE SOURCE BYTE
00380 ;
00390 ; SOURCE BANK = 1 (AERCO)
00400 ; SOURCE BANK = 0 (OTHER)
```

```
00410 ;
00420 ;
00430 XFER2 LD A,1 ; SOURCE BNK
00440 OUT (244),A  ; ENABLE IT
00450 LD A,(DE) ; LOAD S. VALUE
00460 EX AF,AF' ; SAVE IT
00470 ;
00480 ;-------------------------
00490 ;
00500 ; NEXT, ENABLE THE DEST.
00510 ;  BANK, AND STORE THE
00520 ;  SOURCE BYTE THERE
00530 ;
00540 ; DEST BANK = 225 (AERCO)
00550 ; DEST BANK = 224 (OTHER)
00560 ;
00570 LD A,225 ; DESTINATION
00580 OUT (244),A ; ENABLE IT
00590 LD A,(HL) ; LOAD D. VALUE
00600 EX AF,AF' ; SWAP VALUES
00610 LD (HL),A ; STORE DEST.
00620 ;
00630 ; ..... ENABLE THE SOURCE
00640 ;  BANK, AND STORE THE
00650 ;  DEST. BYTE THERE
00660 ;
00670 LD A,1       ; SOURCE BANK
00680 OUT (244),A ; ENABLE IT
00690 EX AF,AF' ; SWAP VALUES
00700 LD (DE),A ; STORE S. BYTE
00710 EX AF,AF' ; RESTORE AF
00720 ;-------------------------
00730 ;
00740 ; INCREMENT THE SOURCE
00750 ;  AND DEST. ADDRESSES.
00760 ;  CHECK TO SEE IF DONE.
00770 ;
00780 INC DE ; SOURCE + 1
00790 INC HL ; DEST. + 1
00800 DEC BC ; LENGTH - 1
00810 LD A,B ; "B" INTO ACC.
00820 OR C   ; SUM WITH "C"
00830 JR NZ,XFER2 ; "BC" = 0 ?
00840 LD A,1 ; RESTORE S. BANK
00850 OUT (244),A  ; ENABLE IT
00860 RET          ; ALL DONE
```

LISTING C

```
00010 ;        XFER_3
00020 ; ------------------------
00030 ;       PRACTICAL
00040 ;     BANK-SWITCHING
00050 ;
00060 ;  ++++++++++++++++++++++
00070 ;  + THIS SAMPLE PROG. +
00080 ;  + WILL MERGE 24064  +
00090 ;  + BYTES FROM MEMORY +
00100 ;  + ADDRESS 41300 OF  +
00110 ;  + STD RAM AND 41300 +
00120 ;  + OF THE DOCK BANK. +
00130 ;  ++++++++++++++++++++++
00140 ;
00150 ; (c) S D LEMKE     1988
00160 ;
00170 ; LEMKE SOFTWARE DEVELOP.
00180 ;      2144 WHITE OAK
00190 ;    WICHITA, KS.  67207
00200 ; ------------------------
00210 ; ------------------------
00220 ;
00230 ; GETTING STARTED...
00240 ;  SET SOURCE, DEST, AND
00250 ;  NUMBER OF BYTES
00260 ;
00270 ORG 39000    ; CODE ADD.
00280 ;
00290 LD HL,41300 ; DEST. ADD.
00300 LD DE,41300 ; SOURCE ADD.
00310 LD BC,24064 ; LENGTH
00320 ;
00330 ;-------------------------
00340 ;
00350 ; BEGIN BY ENABLING THE
00360 ;  SOURCE BANK AND SAVING
00370 ;  THE SOURCE BYTE
00380 ;
00390 ; SOURCE BANK = 1 (AERCO)
00400 ; SOURCE BANK = 0 (OTHER)
00410 ;
00420 ;
00430 XFER3 LD A,1 ; SOURCE BNK
00440 OUT (244),A  ; ENABLE IT
00450 LD A,(DE) ; LOAD S. VALUE
00460 PUSH AF   ; SAVE IT
00470 ;
00480 ;-------------------------
00490 ;
00500 ; NEXT, ENABLE THE DEST.
00510 ;  BANK, AND STORE THE
00520 ;  SOURCE BYTE THERE
00530 ;
00540 ; DEST BANK = 225 (AERCO)
00550 ; DEST BANK = 224 (OTHER)
00560 ;
00570 LD A,225 ; DESTINATION
00580 OUT (244),A ; ENABLE IT
00590 POP AF   ; RECALL S VALUE
00600 OR (HL) ; "OR" DEST VALUE
00610 LD (HL),A ; STORE DEST.
00620 PUSH AF ; SAVE SUM VALUE
00630 ;
00640 ; ..... ENABLE THE SOURCE
00650 ;  BANK, AND STORE THE
00660 ;  "SUM" THERE
00670 ;
00680 LD A,1       ; SOURCE BANK
00690 OUT (244),A ; ENABLE IT
00700 POP AF    ; RECALL "SUM"
00710 LD (DE),A ; STORE S. BYTE
00720 ;-------------------------
00730 ;
00740 ; INCREMENT THE SOURCE
00750 ;  AND DEST. ADDRESSES.
00760 ;  CHECK TO SEE IF DONE.
00770 ;
00780 INC DE ; SOURCE + 1
00790 INC HL ; DEST. + 1
00800 DEC BC ; LENGTH - 1
00810 LD A,B ; "B" INTO ACC.
00820 OR C   ; SUM WITH "C"
00830 JR NZ,XFER3 ; "BC" = 0 ?
00840 LD A,1 ; RESTORE S. BANK
00850 OUT (244),A  ; ENABLE IT
00860 RET          ; ALL DONE
```


AFR SOFTWARE ®

Presents:

Powerful And Inexpensive
Business Software
For "Timex-Sinclair"
Computers

WORD PROCESSING

T/S-TEXT 2000 ..... $19.95
ZX-TEXT $19.95

SPREADSHEET CALCULATOR

T/S-CALC 2000 ..... $19.95
ZX-CALC $19.95

CYCLE ACCOUNTING

T/S-ZX Financial Report Generator $29.95
Printout Of Same $13.00

APPOINTMENT SCHEDULER

T/S-CALENDAR 2000 ..... $19.95
ZX-CALENDAR $19.95

Send S.A.S.E. For Free Catalog
Or Check Or Money Order To:
A.F.R. SOFTWARE
1605 Pennsylvania Ave.
No. 204
Miami Beach, FL 33139
(305) 531-6464
"FLORIDIANS ADD SALES TAX"
Dealer Inquires Invited

XFER_2

```
39000 > 33    > ld hl,NN
39001 > 84      <41300>
39002 > 161
39003 > 17    > ld de,NN
39004 > 84      <41300>
39005 > 161
39006 > 1     > ld bc,NN
39007 > 0       <24064>
39008 > 94
39009 > 62    > ld a,N
39010 > 1       <1>
39011 > 211   > out (N),a
39012 > 244     <244>
39013 > 26    > ld a,(de)
39014 > 8     > ex af,af'
39015 > 62    > ld a,N
39016 > 225     <225>
39017 > 211   > out (N),a
39018 > 244     <244>
39019 > 126   > ld a,(hl)
39020 > 8     > ex af,af'
39021 > 119   > ld (hl),a
39022 > 62    > ld a,N
39023 > 1       <1>
39024 > 211   > out (N),a
39025 > 244     <244>
39026 > 8     > ex af,af'
39027 > 18    > ld (de),a
39028 > 8     > ex af,af'
39029 > 19    > inc de
39030 > 35    > inc hl
39031 > 11    > dec bc
39032 > 120   > ld a,b
39033 > 177   > or c
39034 > 32    > jr nz,DIS
39035 > 229     <229>
39036 > 62    > ld a,N
39037 > 1       <1>
39038 > 211   > out (N),a
39039 > 244     <244>
39040 > 201   > ret
```

XFER_3

```
39000 > 33    > ld hl,NN
39001 > 84      <41300>
39002 > 161
39003 > 17    > ld de,NN
39004 > 84      <41300>
39005 > 161
39006 > 1     > ld bc,NN
39007 > 0       <24064>
39008 > 94
39009 > 62    > ld a,N
39010 > 1       <1>
39011 > 211   > out (N),a
39012 > 244     <244>
39013 > 26    > ld a,(de)
39014 > 245   > push af
39015 > 62    > ld a,N
39016 > 225     <225>
39017 > 211   > out (N),a
39018 > 244     <244>
39019 > 241   > pop af
39020 > 182   > or (hl)
39021 > 119   > ld (hl),a
39022 > 245   > push af
39023 > 62    > ld a,N
39024 > 1       <1>
39025 > 211   > out (N),a
39026 > 244     <244>
39027 > 241   > pop af
39028 > 18    > ld (de),a
39029 > 19    > inc de
39030 > 35    > inc hl
39031 > 11    > dec bc
39032 > 120   > ld a,b
39033 > 177   > or c
39034 > 32    > jr nz,DIS
39035 > 229     <229>
39036 > 62    > ld a,N
39037 > 1       <1>
39038 > 211   > out (N),a
39039 > 244     <244>
39040 > 201   > ret
```

AERCO FD-68 DISK SYSTEM

AERCO MERGE FUNCTION

by Larry Zunk

Larry Zunk of Zunk Custom Electronics (4800 East Cedar lane, Norman, Oklahoma 73071), shares a few routines for the AERCO FD-68 disk drive system. Larry has programmed a powerful software package for the FD-68 called "CADZ" (watch for review in upcoming issue, or write to the above address for further details).

This is a tip for all Aerco FD-68 users. The basic MERGE function has never been available, but a simulated merge can be accomplished.

Rule #1: line numbers must be consecutive. Rule #2: get rid of all variables. Rule #3: is make sure there is enough room for both listings.

It works like this. In the immediate mode, type:

```
CAT "first program.BAS",
CLEAR 65535
POKE 23635,PEAK 23627     (POKE prog,vars)
POKE 23636,PEAK 23628
CAT "second program.BAS",
```

(NOTE: Execution will stop here, and all you will see is the second listing.) Then also in the immediate mode, type:

```
POKE 23635,86
POKE 23636,104            (POKE prog,26710)
LIST
```

SUPER DETAILED DISK DIRECTORY

by Larry Zunk

The following program listings are for the AERCO FD-68 Disk Drive System. I must give credit to Mowgli Assor for his user tips in the SEP/OCT '87 issue of TDM. His machine code routines are the heart of these programs.

Listing 1 is a 32 column detailed disk directory. Listing 2 is a 64 column detailed directory for use with the Advanced Video Modes software by BEAVER COMPUTER PRODUCTS. Either version can be run in BASIC or compiled with the TIMACHINE compiler by NOVELSOFT.

The directory gives file name, type, length, location, auto start line numbers, length of variables, active chunks, and tracks allocated for each file. Although the information is the same, I prefer the 64 column version because everything fits on one line and it gives a much cleaner screen display.

My system is double-sided double-density, so some changes to the machine code in the data statement may be required for other systems. The disk directory is on track 0 sectors 2 and 3. The code sets the start of the buffer at E290h (58000 decimal) and reads track 0 sector 2, then increments the buffer address held in the HL register by 512 bytes. Then it sets the B register to 3 and reads sector 3. The AERCO user manual states that a single-density system has 256 bytes per sector and a double-density system has 512 bytes per sector (I assume that a quad-density system would have 1024 bytes per sector). This is what will need to be changed for systems other than DD/DD.

LISTING 1

```
  10 REM !USR 60000
  20 REM !INT +LEN,LNLOC,VAR,BUF
,TYP,LOC,F,E
  30 REM !LEN P$<=27
  40 REM ! LIST
  50 REM ! OPEN #
  60 FOR F=59967 TO 59999: READ
E: POKE F,E: NEXT f: RANDOMIZE U
SR 59967
  70 DATA INT 62,3,211,244,205,6
6,53,1,2,0,33,144,226,205,86,53,
1,3,0,33,144,228,205,86,53,205,1
30,53,62,1,211,244,201
  80 LET BUF=58032
  90 LET P$="BASDATCHRBINSCRAROL
ROBUTVAR"
 100 CLS : PRINT INVERSE 1;"  FI
LE              LINE# VARS   NA
ME    TYPE BYTES LOC    CHUNK"; I
NVERSE 0;AT 0,21; OVER 1;"_____
_____";AT 2,0;
 110 LET LOC=BUF: LET TYP=PEEK L
OC*3+1: LET LOC=LOC+1: LET E=LOC
+10
 120 IF PEEK LOC<>0 AND LOC<E TH
EN PRINT CHR$ PEEK LOC;: LET LOC
=LOC+1: GO TO 120
 130 LET LOC=E
 140 GO SUB 300: LET LEN=E
 150 GO SUB 300: LET LNLOC=E
 160 GO SUB 300: LET VAR=E
 170 IF TYP=13 THEN LET LEN=6912
 180 PRINT TAB 10;".";P$(TYP TO
TYP+2);TAB 15;LEN;TAB 21;
 190 IF TYP=1 AND LNLOC<10000 TH
EN PRINT "L";LNLOC;TAB 27;LEN-VA
R;
 200 IF TYP=10 THEN PRINT LNLOC;
TAB 27;INT (VAR/256)-2;
 210 IF TYP=13 THEN PRINT 16384;
 220 PRINT TAB 0;"*";
 230 IF PEEK LOC<>0 THEN PRINT P
EEK LOC;"*";: LET LOC=LOC+1: GO
TO 230
 240 POKE 23689,PEEK 23689+1: PR
INT OVER 1;"_____
_____";
 250 LET BUF=BUF+32
 260 PRINT TAB 0;
 270 IF PEEK 23689=2 THEN PRINT
#1;AT 1,0; INVERSE 1;"     HIT E
NTER TO CONTINUE      "; INVERSE
0: PAUSE 0: GO TO 100
 280 IF LOC>58992 THEN STOP
 290 GO TO 110
 300 LET E=PEEK LOC+256*PEEK (LO
C+1): LET LOC=LOC+2: RETURN
 310 REM ! CLOSE #
9999 ERASE "32COLDIR.BAS",: MOVE
 "32COLDIR.BAS",
```

LISTING 2

```
  10 REM !USR 50000
  20 REM !INT +LOOP,LNLOC,VARS,L
EN,LOC,B(),N,F
  30 REM !LEN P$<=27
  40 REM ! LIST
  50 REM ! OPEN #
```

Listing Continued Next Page

```
  55 DATA INT 62,3,211,244,205,6
6,53,1,2,0,33,144,226,205,86,53,
1,3,0,33,144,228,205,86,53,205,1
30,53,62,1,211,244,201
  56 FOR F=59967 TO 59999: READ
N: POKE F,N: NEXT f: RANDOMIZE U
SR 59967
  60 RANDOMIZE USR 61450: POKE 2
3578,21: PRINT #4;CHR$ 3+CHR$ 62
;CHR$ 0
  70 LET LOOP=16: LET LOC=58031:
 DIM B(25)
  80 LET P$="BASDATCHRBINSCRAROL
ROBUTVAR"
  90 PRINT AT 0,0; INVERSE 1;" F
ile Name Typ:Bytes+Vars :Line:Ad
dr:Chunks:Tracks**************";
 INVERSE 0
 100 FOR F=1 TO LOOP
 110 FOR N=1 TO 25: LET B(N)=PEE
K (LOC+N): NEXT N
 120 FOR N=2 TO 11: IF b(n)>31 T
HEN PRINT AT f,n-2;CHR$ b(n);
 130 NEXT n: LET VARS=B(1)+3+1:
PRINT TAB 10;".";P$(VARS TO VARS
+2);TAB 15;
 140 LET LEN=B(12)+256*B(13)
 150 LET LNLOC=B(14)+256*B(15)
 160 LET VARS=B(16)+256*B(17)
 170 IF B(1)=0 THEN PRINT VARS;T
AB 20;"+";LEN-VARS;TAB 27;(LNLOC
 AND LNLOC<10000);
 180 IF B(1)=3 OR B(1)=8 THEN PR
INT LEN;TAB 32;LNLOC;TAB 38;: IF
 b(1)=3 THEN PRINT B(17)-2;
 190 IF B(1)=4 THEN PRINT 6912;T
AB 32;16384;
 200 PRINT TAB 44;: FOR N=18 TO
25: IF B(N)>0 THEN PRINT (B(N)-(
87 AND B(N)>99));" ";
 210 NEXT N
 220 LET LOC=LOC+32
 230 NEXT f
 240 PRINT ' INVERSE 1;"    HIT
 [ENTER]  OR  [FIRE BUTTON] TO
CONTINUE    "; INVERSE 0
 250 IF INKEY$="" AND STICK (2,1
)=0 THEN GO TO 250
 260 PRINT CHR$ 0;: IF LOOP=15 T
HEN PRINT CHR$ 3+CHR$ 0;CHR$ 0:
POKE 23578,6: STOP
 270 LET LOOP=15: GO TO 90
 280 REM ! CLOSE #
9999 OUT 244,1: ERASE "newdir.ba
s",: MOVE "newdir.bas",
```

SYNX

by Jack Dohany

SYNX is a 46-byte relocatable MC routine for AERCO disk drive system users. It is given into the "public domain". SYNX is short for "Syntax Checker Switch". This routine allows you to turn off the BASIC syntax checker when writing or editing BASIC lines...and to turn it back on. Syntax checking during program execution remains in effect.

Why? Perhaps you may want to write a BASIC program that can be used on a non-AERCO disk system (perhaps for ALL disk systems). You may want to write a line like this:

500 SAVE *"TEST": REM for Zebra disk

Well, you can't write it because it will fail syntax checking. But with the syntax checker turned off, you can write it.

SYNX works by changing some locations in the BASIC operating system...normally in ROM, but in RAM with AERCO disk. A more detailed explanation is beyond the scope of this article.

Assuming you have the code on disk or tape as .BIN or CODE file, it can be loaded wherever there's no conflict with other software. Let us say you want to load it at Loc 64000: CAT "SYNX.bin",64000 or LOAD "SYNX" CODE 64000 will do it.

To SAVE the code: MOVE "SYNX.bin",64000,46 or SAVE "SYNX" CODE 64000,46

To use SYNX (assuming the code is at 64000): RANDOMIZE USER 64000 turns the syntax checker OFF. RANDOMIZE USER 64002 turns the syntax checker back ON.

Here is the SYNX code as a decimal listing, ready to be POKEd into memory however you wish:

64000	24	64020	24	64040	0
64001	27	64021	221	64041	243
64002	24	64022	197	64042	237
64003	18	64023	225	64043	176
64004	0	64024	1	64044	251
64005	0	64025	11	64045	201
64006	0	64026	0		
64007	253	64027	24		
64008	54	64028	5		
64009	0	64029	197		
64010	255	64030	255		
64011	0	64031	1		
64012	0	64032	4		
64013	205	64033	0		
64014	13	64034	9		
64015	13	64035	17		
64016	253	64036	76		
64017	54	64037	14		
64018	0	64038	1		
64019	255	64039	9		

Of course, if you get a single number wrong, the computer is likely to crash when you attempt to use SYNX. So it is a good idea to SAVE before you test it out.

TIMACHINE ON AERCO DISK

by Carl Green

Here is a hint for putting TIMACHINE (Novelsoft) on the AERCO disk drive system. I found the FD-68 does not like:

1) equations
2) VAL "____"
3) more than one period (.)
4) scientific notations (eg: 6e4)

in the CAT and MOVE statements. Try the following for TIMACHINE:

1) LOAD the whole program from tape.
2) Move the cursor to the right of the quotation marks.
3) DELETE the quotation marks.
4) Press STOP once, then ENTER twice.
5) Change line 8070 to read:

```
8070 PRINT AT 19,0: MOVE "TSTIME
.BAS",9997: MOVE "TSTMLGO.BIN",6
0614,4922: MOVE "TSSETUP.BIN",60
000,83: MOVE "TSTIME2.BIN",26688
,11458
```

6) Change line 9997 to read:

```
9997 CLEAR 60613: INK 6: PAPER 6
: BORDER 6: CLS : PRINT AT 19,0:
 CAT "TSTMLGO.BIN",: RANDOMIZE U
SR 60614: INK 6: PRINT AT 19,0:
CAT "TSSETUP.BIN",: RANDOMIZE 11
458: LET X=USR 60000: PRINT AT 1
19,0: INK 6: CAT "TSTIME2.BIN",:
INK 0: GO TO 8000
```

7) OUT 244,1
8) GOTO 8000
9) Put formatted disk in.
10) At the "BACK UP?" option, press "Y".

TIMACHINE should now be on disk with the backup option operational.

LARKEN 2068 DISK DRIVE SYSTEM

CASSETTE TO LARKEN DISK

by Gaylen W. Bench

From the time I first bought my Timex Sinclair 1000 for a close-out price of $29.95 (around 1982), I have always dreamed of the day when I could have a "complete" computer system.

I moved one step closer with the purchase of a used TS2068 in November of 1986. But I was still stuck with using cassette tapes.

Next I moved up to A&J Microdrives (for my TS2068). It was so much faster and easier to use. Over the next few weeks I spent a lot of time converting all of my cassette software to the A&J. I was happy with the A&J for almost six months...maybe even a little longer.

Then one day I was over at a friend's home and saw how nice his computer worked with a disk drive system. Now that would really be something to have a disk system for my hard-working Timex Sinclair. But, it was just too expensive to add one to my computer system.

Then along came Mr. Larry Kenny of LARKEN ELECTRONICS, and his floppy disk drive system for the TS2068. The cost was low enough to give it a try. Now about a year later, my present system includes the used 2068, a Larken disk drive system, a dual Amdek Amdisk III, one Quad 5 1/4" drive, an RX-80 Epson printer, an Aerco printer interface, a TS2040 printer, a 2050 modem, a green screen monitor, and lots of software. If I had the money that I have spent on this system over the last six years, I could buy a "basic" PC compatible, with NO software or peripheral hardware. I'm going to stick with my Sinclair.

Now that you know a little of the history of my hardware system, maybe I can help you convert cassette (or A&J Micro Drive cartridge) software to the Larken Disk Drive system. During this last year, I have, with the help and advise of some fellow Sinclair users, converted the following software to the Larken:

Tasword II, Pro/File 2068, Pro/File +5, VuFile, VuCalc, Timemachine, Pixel Print, Zeus, Loader V, Jet Set Willy, Voice Chess, Greeting Card Designer, Banner Designer, Personal Accountant, Kruncher, programs on the original tape supplied with the 2068, and many other pieces of software taken from a variety of sources.

Changing all of this software to the Larken DOS (Disk Operating System) has one thing in common: the changes made in each program is a modification of the BASIC save and load commands, to save the program, to load and save code, to load and save data strings, to load and save screen strings. Refer to the Larken operations manual for instructions on how the "RANDOMIZE USR 100" is used before each load or save command in your BASIC program. The following listing is an example of how these changes were made in TASWORD II (2068 word processor):

```
  15 RANDOMIZE USR 100: OPEN #4,
"dd": POKE VAL "23609",VAL "2":
CLEAR VAL "33279": GO SUB VAL "4
000": BORDER VAL "2": PAPER VAL
"4": INK VAL "9": PRINT #4: LOAD
 "TW.Ct"CODE : CLS : LET a=USR V
AL "59081": GO TO VAL "10"
  25 GO SUB VAL "4000": PRINT AT
 VAL "2",VAL "0";"print text fil
e";TAB VAL "31";"p"
  28 PRINT : PRINT "save text fi
le";TAB VAL "31";"s"
  30 PRINT : PRINT "load text fi
le";TAB VAL "31";"j"
  35 PRINT : PRINT "merge text f
ile";TAB VAL "31";"m"
  40 PRINT : PRINT "return to te
xt file";TAB VAL "31";"y"
  45 PRINT : PRINT "define graph
ics/printer";TAB VAL "31";"g"
  50 PRINT : PRINT "save tasword
";TAB VAL "31";"t"
  55 PRINT : PRINT "into Basic";
TAB VAL "31";"b"
  60 PRINT : PRINT "DIRECTORY";T
AB 31;"d"
  70 PRINT AT VAL "20",VAL "0";"
                press key
"
 110 IF b=VAL "115" THEN LET i=V
AL "6"
 120 IF b=VAL "106" THEN LET i=V
AL "8"
 125 IF b=VAL "116" THEN LET i=V
AL "16"
 130 IF b=VAL "112" THEN LET i=V
AL "4"
 140 IF b=VAL "121" THEN LET i=V
AL "12"
 150 IF b=VAL "109" THEN LET i=V
AL "10"
 160 IF b=VAL "103" THEN LET i=V
AL "14"
 170 IF b=VAL "98" THEN LET i=VA
L "18"
 175 IF b=VAL "100" THEN GO TO V
AL "9970"
 700 CLS : LET i=VAL "8": GO SUB
 VAL "800": GO SUB VAL "9900": L
ET a$="TW": PRINT #4: SAVE a$+".
Bt" LINE VAL "15"
 710 PRINT #4: SAVE a$+".Ct"CODE
 VAL "54784",VAL "10751": GO TO
VAL "25"
1030 LET i=VAL "12": GO SUB VAL
"800": GO SUB VAL "9900": PRINT
#4: SAVE a$+".Ct"CODE b,a: CLS
2030 LET b=FN p(VAL "62216"): GO
 SUB VAL "9900": PRINT #4: LOAD
 a$+".Ct"CODE (a+b),((FN p(VAL "6
2221")+VAL "22")*VAL "64"-a): GO
 TO VAL "10"
9900 INPUT "Drive: 0, 1, 2, 3  "
;Dr: PRINT #4: GO TO Dr: RETURN
9970 CLS : GO SUB VAL "9900": PR
INT #4: CAT "",: PAUSE 0: CLS :
GO TO VAL "25"
```

Line 15 sets up the PRINT #4 command that will be used before each save and load command in the program. Line 60 adds a disk directory choice to the tasword menu. Line 175 is the IF..THEN statement used by the menu to call Line 9970. Line 9970 calls the GO SUB Line 9900 which gives you a choice of which drive you want and then returns to do the catalog of that chosen drive. After the directory is done, the program will return to the main menu. Please note that the drive selection GO SUB Line 9900 is also called by both the load and save lines of the program. Lines 700-710 are the save lines that will save Tasword II to disk. Line 1030 is the save line that will save all files (letters or documents) to disk. Line 2030 is the load line that will load your chosen file from the disk to Tasword.

I know that this is a brief description of how these lines are changed. But there is one area of concern that has to be dealt with in making these changes. That area is the memory spaces for the basic program. You will note in Line 15, that RAMTOP is lowered to 33279, and then the Tasword code is loaded above that. The basic program cannot be written above that address. If you change that address...well, that's another complete article.

There are a number of ways that the original basic program can be changed to free up memory space to allow for these changes. If you are going to dedicate the program to disk use only, you can go in and DELETE the cassette VERIFY routines. The Larken system uses the VERIFY command as a "disk check" command (refer to the Larken manual for further details). You can also change or eliminate any prompts that are displayed on the screen concerning the loading, saving, or verifying of cassette tapes.

One more way to free up program memory space is the use of the KRUNCHER program (written by Syd Wyncoop and available from RMG Enterprises). KRUNCHER will "modify" the basic listing. It places all numbers found in the listing inside VAL "" statements, and replaces the number 0 with NOT PI, 1 with SGN PI, and 3 with INT PI. These tokens have the same value as the number. There is one problem in using the KRUNCHER program: it does not change negative numbers correctly. A -250 will be changed to VAL "-250", which causes a syntax error in the basic program. The way to correct this is, before you "krunch" the program, list it out and put any negative number inside brackets (-250 will be -(250)). The KRUNCHER program will now convert this number correctly.

There are some other problems encountered when converting cassette software to disk. One of the first ones you may come across, is the program (or file) name. Cassette allows a name to be ten characters long. Larken allows up to six characters plus an "extension". You can see how the program (file) name and extension are used, by looking at the load and save lines of the Tasword listing example (please refer to the Larken manual for rules concerning the use of extensions). Some basic programs include a "test statement", testing for the cassette ten character name length. These statements will have to be changed to conform to the Larken file name length.

I hope that I have been able to help some of you with the change from cassette to Larken disk. If any of you have suggestions on how these changes can be made easier than I have listed, please feel free to contact me. I'm Gaylen W. Bench, and my address is: 900 N.W. Mawcrest Dr. #110, Gresham, Oregon 97030. You can also contact me via CompuServe (ID# 73720,755), or on the RMG BBS (503-656-8072; settings 8/1/N). I will gladly pass any updated information to TDM readers in a future article.

The information provided in this article is a long way from being complete on how to convert every program to disk. If any of you would like assistance in making these changes, please contact me, and we will work out some arrangement on how to get the job done.

One more important find. If any of you have attempted to put a choice in your basic program to switch between the Timex 2040 printer and a full-size printer, you will run into a problem using the Larken system. For some reason the Larken system over-writes the 2068's print buffer. You can find out if you have this problem by doing an LLIST to the 2040 printer. If the first thing printed is garbage, then you have the problem. I tried for about a year to find a solution. Whenever I would use the PRO/FILE +5 program, I could not switch back to the 2040 printer after using the full-size printer...the program would "crash". The correction to this problem is so simple! When you change back to the 2040, the first thing you need to do is a simple "LPRINT". This clears the 2068 print buffer and now you can print with the 2040 without any trouble.

PASCAL DISK HANDLER FOR THE LARKEN

Article by David Solly
Programs by David Solly and Larry Kenny

Access to the Larken Disk Drive System from within a compiled HiSoft(TM) Pascal program is now possible thanks to the procedures developed by David Solly and Larry Kenny. This article demonstrates how to install these procedures and gives a practical demonstration on how they may be used within a simple directory program. The procedures described in this article are valid for both the Timex Sinclair 2068 and the ZX Spectrum versions of HiSoft Pascal.

I have owned HiSoft Pascal, (henceforth Pascal), for the Timex Sinclair 2068 and the ZX Spectrum for a number of years now but the drawback with Pascal, as it is implemented on Sinclair computers, has been that there was no way to access a DOS from within a compiled program. It seemed ridiculous to put the effort into creating programs which ran like machine code once the source code was debugged, compiled and transfered to disk only to be forced to go back to a plodding tape operating system, (henceforth TOS), when I needed to save or enter data from within the compiled program. Finally I sat down and did some serious study of the problem, some head scratching, some question asking (especially of Larry Kenny, who is the creator of the Larken DOS, and Ken Schieman) and some experimentation. The result of this is the creation of the Pascal disk handler described below.

The needs of LKDOS are the same as for the TOS. You are required to provide the name of your program, the starting address of the program and the lenght of the program. This is exactly what happens when you type within Basic:

```
SAVE "Myprogram" CODE start, length.
```

Pascal also passes the same information to the TOS when it saves out a variable through the use of the procedure:

```
TOUT ('Myprogram', ADDR (the name of the variable being saved), SIZE (the name of the variable being saved));
```

The crux of the problem is that, although both languages generate identical information, LKDOS can extract it only from within the Basic environment. Therefore, additional programming is need in order to transfer the save or load parameters to LKDOS from within the Pascal environment.

The first steps towards transfering the required information form Pascal to LKDOS are to store the name, the start address, and the length of the variable to be saved or loaded in a safe area of the RAM. This is a simple matter thanks to the very versatile POKE() function in Pascal. Unlike the POKE command available in Basic, this Pascal function can DOKE a number or POKE the contents of a whole character array starting at a given address. The first fifteen bytes of the printer buffer proved to be the safest area to which all the transfers can be achieved with the following lines:

```
POKE (23300, 'Myprogram');

POKE (23311, ADDR (name of variable));

and

POKE (23313, SIZE (name of variable));
```

You can see these lines in a modified form in the demonstration program within the procedures GETNAME, DISKSAVE and DISKLOAD.

Upon completion of these steps the next step is to copy the information from its storage place in the RAM to the appropriate routines in the LKDOS. Again to HiSoft includes within their implementation of Pascal the procedure INLINE() wich allows us to embed Z80 machine code within a Pascal procedure to accomplish our ends.

The following is a disassembly of the machine code used in the procedure DOUT

```
00010 NAME      EQU 23300
00020 PROGNM    EQU 8226
00030 TEMP4     EQU 8243
00040 TEMP2     EQU 8241
00050 NMIF      EQU 8194
00060 ADDR      EQU 23311
00070 SIZE      EQU 23313
00080 SV1       EQU 0204
00090 SV2       EQU 0207
00100           ORG 40000
00110           DI                ; Disable interrupt
00120           CALL 98           ; Turn on LKDOS cartridge
00130           LD HL,NAME        ; Transfer file name
00140           LD DE,PROGNM      ; to prognm
00150           LD BC,09
00160 MOVE      LD A,(HL)         ; Loop to catch any
00170           CP 0              ; occurance of CHR$ 0
00180           JR NZ,NZERO
00190           LD (HL),32        ; & replace with a space
00200 NZERO     LDI
00210           LD A,B
00220           OR C
00230           JR NZ,MOVE
00240           LD A,11
00250           LD (NMIF),A
00260           CALL SV1          ; Save name
00270           LD HL,(ADDR)      ; Retrieve start address
00280           LD (TEMP4),HL     ; Set start address
00290           LD HL,(SIZE)      ; Retrieve length of save
00300           LD (TEMP2),HL     ; Set length of save
00310           CALL SV2          ; Second save data
00320           LD A,(100)        ; Exit cartridge
00330           EI                ; Enable interrupt
```

(Program provided by Larry Kenny of Larken Electronics, #2 Navan, Ontario, Canada K4B 1H9, Tel: (613)-835-2680) (Listing made using Zeus Assembler)

The procedure DIN contains the same code except that SV1 and SV2 are replaced with LD1 EQU 198 and LD2 EQU 201. The MOVE loop insures that the file name is padded out with the correct number of spaces so that it totals nine characters and spaces otherwise the file name may be corrupted and fail to reload.

The following listing demonstrates how all the procedures described above are used within a complete Pascal program. The object of the program is to create a simple telephone directory which will allow you to store ten names and numbers, read the information stored in the directory, and read and write the information stored in the directory to disk using the Larken LKDOS. There is also a summation of this article contained in the procedure SONGANDDANCE.

Pascal source code listing

```
 10 {PROGRAM BY:                }
 20 {DAVID SOLLY                }
 30 {1402-1545 ALTA VISTA DRIVE}
 40 {OTTAWA, ONTARIO            }
 50 {CANADA K1G 3P4             }
 60 {TEL: (613)-731-2120        }
 70
 80
 90 {THIS PORGRAM IS FOR        }
100 {DEMONSTRATING HOW THE      }
110 {LARKEN DISK DRIVE MAY BE   }
120 {ACCESSED FOR STORING AND   }
130 {RETRIEVING DATA WITHIN A   }
140 {HISOFT(TM) PASCAL PROGRAM.}
150 {THE PROCEDURES DESCRIBED   }
160 {IN THIS PROGRAM ARE VALID  }
170 {FOR BOTH THE ZX SPECTRUM   }
180 {AND  THE TIMEX SINCLAIR    }
190 {2068 VERSIONS OF HISOFT    }
200 {PASCL.                     }
210
220
230 PROGRAM LARKENDEMO;
240
250 CONST
260   LENGTH = 10;
270   PN = 23300; {ADDRESS WHERE THE NAME FOR THE SAVE/LOAD}
280               {ROUTINES IS STORED}
290   PB = 23311; {CONTAINS THE ADDRESS WHERE THE DATA BEGINS}
300   PS = 23313; {CONTAINS THE NUMBER OF BYTES USED BY THE DATA}
310   CAPSLOCK = 23658; {ADDRESS OF THE CAP SHIFT LOCK CONTROL}
320   ON  = 8; {SWITCHES CAPSLOCK ON}
330   OFF = 0; {SWITCHES CAPSLOCK OFF}
340
350
360 TYPE
370   ENTRY = RECORD
380             NAME : ARRAY [1..10] OF CHAR;
390             NUMBER : ARRAY [1..10] OF CHAR;
400           END;
410
420
430 VAR
440   DIRECTORY : ARRAY [1..LENGTH] OF ENTRY;
450   I : INTEGER;
460   FINISHED : BOOLEAN;
470   ANS : CHAR;
480
490
500
510
520 PROCEDURE DOUT; {INVOKES LKDOS WRITE}
530   BEGIN
540     INLINE  (#F3, #CD, #62, #00, #21, #04, #5B, #11,
550              #22, #20, #01, #09, #00, #7E, #FE, #00,
560              #20, #02, #36, #20, #ED, #A0, #78, #B1,
570              #20, #F3, #3E, #0B, #32, #02, #20, #CD,
580              #CC, #00, #2A, #0F, #5B, #22, #33, #20,
590              #2A, #11, #5B, #22, #31, #20, #CD, #CF,
600              #00, #3A, #64, #00, #FB);
610 END;
620
630
640 PROCEDURE DIN; {INVOKES LKDOS READ}
650   BEGIN
660     INLINE  (#F3, #CD, #62, #00, #21, #04, #5B, #11,
670              #22, #20, #01, #09, #00, #7E, #FE, #00,
680              #20, #02, #36, #20, #ED, #A0, #78, #B1,
690              #20, #F3, #3E, #0B, #32, #02, #20, #CD,
700              #C6, #00, #2A, #0F, #5B, #22, #33, #20,
710              #2A, #11, #5B, #22, #31, #20, #CD, #C9,
720              #00, #3A, #64, #00, #FB);
730 END;
740
750
760 PROCEDURE GETNAME;
770 VAR
780   PROGNM : ARRAY [1..9] OF CHAR;
790
800   BEGIN
810     WRITELN;
820     POKE (CAPSLOCK, OFF);
830     WRITELN ('NAME FOR DISK OPERATION? ');
840     WRITELN;
850     WRITE ('9 CHARACTERS MAXIMUM:  ');
860     READLN;
870     READLN (PROGNM);
880     POKE (PN, PROGNM); {STORES THE NAME IN RAM}
890   END;
900
910
920 PROCEDURE DISKSAVE;
930 BEGIN
940   PAGE;
950   WRITELN;
960   WRITELN ('LARKEN DISK SAVE ROUTINE');
970   WRITELN;
980   GETNAME;
990   POKE (PB, ADDR (DIRECTORY)); {START ADDRESS FROM WHERE THE}
1000                               {DATA IS TO BE SAVED}
1010  POKE (PS, SIZE (DIRECTORY)); {NUMBER OF DATA BYTES TO SAVE}
1020  DOUT;
1030  WRITELN ('SAVED!');
1040  FOR I := 1 TO 100 DO {PAUSE LOOP}
1050 END;
1060
```

```
1070
1080 PROCEDURE DISKLOAD;
1090 BEGIN
1100   PAGE;
1110   WRITELN;
1120   WRITELN ('LARKEN DISK LOAD ROUTINE');
1130   WRITELN;
1140   GETNAME;
1150   POKE (PB, ADDR (DIRECTORY)); {START ADDRESS AT WHICH THE}
1160                                {DATA IS TO BE LOADED}
1170   POKE (PS, SIZE (DIRECTORY)); {NUMBER OF DATA BYTES TO LOAD}
1180   DIN;
1190   WRITELN ('LOADED!');
1200   FOR I := 1 TO 100 DO {PAUSE LOOP}
1210 END;
1220
1230
1240 PROCEDURE FILLDIRECTORY;
1250 BEGIN
1260   PAGE;
1270   POKE (CAPSLOCK, OFF);
1280   FOR I := 1 TO LENGTH DO
1290     BEGIN
1300       WITH DIRECTORY [I] DO
1310         BEGIN
1320           WRITELN ('ENTRY NO. ', I, ' OF ', LENGTH);
1330           WRITELN;
1340           WRITE ('NAME PLEASE (10 CHARS) ');
1350           READLN;
1360           READ (NAME);
1370           WRITE ('NUMBER PLEASE (10 CHARS) ');
1380           READLN;
1390           READ (NUMBER)
1400         END
1410      END;
1420     WRITELN ('DIRECTORY FULL ');
1430     WRITELN;
1440     WRITELN ('HIT ANY KEY TO CONTINUE');
1450     READLN
1460 END;
1470

1480
1490 PROCEDURE READDIRECTORY;
1500 BEGIN
1510   PAGE;
1520   POKE (CAPSLOCK, OFF);
1530   FOR I := 1 TO LENGTH DO
1540     BEGIN
1550       WITH DIRECTORY [I] DO
1560         BEGIN
1570           WRITELN (NAME,'      ', NUMBER);
1580           WRITELN;
1590         END
1600     END;
1610    WRITELN ('END OF DIRECTORY');
1620    WRITELN ('HIT ANY KEY TO CONTINUE');
1630    READLN
1640 END;
1650
1660
1670 PROCEDURE SONGANDDANCE;
1680   BEGIN
1690     PAGE;
1700     WRITELN ('Larken Disk Access Routine');
1710     WRITELN ('for');
1720     WRITELN ('HiSoft(TM) Pascal');
1730     WRITELN ('for the');
1740     WRITELN ('ZX Spectrum');
1750     WRITELN ('and the');
1760     WRITELN ('Timex Sinclair 2068');
1770     WRITELN;
1780     WRITELN ('by');
1790     WRITELN ('David Solly');
1800     WRITELN ('and');
1810     WRITELN ('Larry Kenny');
1820     WRITELN;
1830     WRITELN ('Hit any key to continue');
1840     READLN;
1850     PAGE;
1860     WRITELN ('Many ZX Spectrum and TS 2068');
1870     WRITELN ('programers have long wanted to');
1880     WRITELN ('do serious programing in other');
1890     WRITELN ('languages than the resident');
1900     WRITELN ('Sinclair Basic. Although such');
1910     WRITELN ('languages as Forth, Logo, C,');
1920     WRITELN ('Prolog and Pascal have long been');
1930     WRITELN ('available to Sinclair users one');
1940     WRITELN ('of the main drawbacks for');
1950     WRITELN ('serious programing in these');
1960     WRITELN ('languages has been the lack of');
1970     WRITELN ('disk I/O routines. This program');
1980     WRITELN ('will demonstrate how the Larken');
1990     WRITELN ('disk drive system may be');
2000     WRITELN ('accessed for storing and ');
2010     WRITELN ('retrieving data within a');
2020     WRITELN ('HiSoft(TM) Pascal program. The');
2030     WRITELN ('procedures described in this ');
2040     WRITELN;
2050     WRITELN ('Hit any key to continue');
2060     READLN;
2070     PAGE;
2080     WRITELN ('program are valid for both the');
2090     WRITELN ('ZX Spectrum and the Timex');
2100     WRITELN ('Sinclair 2068');
2110     WRITELN;
2120     WRITELN ('The two disk drive procedures');
2130     WRITELN ('are found in the procedures DIN,');
2140     WRITELN ('and DOUT. A third procedure,');
2150     WRITELN ('called GETNAME, supplies the');
2160     WRITELN ('above procedures with a name for');
2170     WRITELN ('storing or retrieving from the');
2180     WRITELN ('disk. All these procedures work');
2190     WRITELN ('in conjunction with the resident');
2200     WRITELN ('procedures ADDR(), SIZE(), and');
2210     WRITELN ('POKE()');
2220     WRITELN;
2230     WRITELN ('The authors hope that these');
2240     WRITELN ('procedures will stimulate Pascal');
2250     WRITELN ('programing for the ZX Spectrum');
2260     WRITELN;
2270     WRITELN ('Hit any key to continue');
2280     READLN;
2290     PAGE;
2300     WRITELN ('and the Timex Sinclair 2068 and');
2310     WRITELN ('encourage other Sinclairests to');
2320     WRITELN ('write disk routines for the');
2330     WRITELN ('other languages mentioned');
2340     WRITELN ('above.');
```


For the TIMEX 2068 and OKIMATE 20

Color Copy

NEW for your 2068!!

COPY A SCREEN$ IN COLOR? YES!
Now a T/S 2068 artist can copy to paper his/her favorite screen$ ..... IN FULL COLOR!! All eight Timex paper & ink colors are accurately reproduced for a full 24 line X 32 column graphics screen dump. The user friendly software provides for two copy sizes in color or in black & white.

HARDWARE REQUIREMENTS? - An OKIMATE 20 printer w/ COMMODORE PLUG 'n PRINT CARTRIDGE are interfaced to a 2068 via a simple COMMODORE serial port emulator circuit; (The same circuit also interfaces a 2068 to a COMMODORE 1520 4-color printer/plotter.) The I/F hardware sells for $14.95 (bareboard only), $20.95 (complete kit), and $30.95 (assembled & tested) - all postpaid.

OTHER SOFTWARE? - The OKIMATE 20 is a many featured 80/136 col. printer, able to print in draft, NLQ, italics, reverse (white on black), underline, super/subscripts, six char. sizes, and do 7 or 24 pin color or b/w graphics. A "patch kit" software program allows CMScript versions 5/5.2 to print to the OKIMATE 20. A hi-mem driver code block is also available for LPRINT/LLISTing to the OKIMATE 20 from BASIC. All software is priced at $8.95 postpaid each and comes with complete user notes.

Send LSASE for additional information and order form to:

♦♦♦ John McMichael ♦♦♦
♦♦♦ 1710 Palmer Drive ♦♦♦
♦♦♦ Laramie, WY 82070 ♦♦♦

```
2350     WRITELN;
2360     WRITELN ('David Solly');
2370     WRITELN ('Larry Kenny');
2380     WRITELN;
2390     WRITELN;
2400     WRITELN ('Hit any key to continue');
2410     READLN
2420   END;
2430
2440
2450 BEGIN {BODY OF THE PROGRAM}
2460   REPEAT
2470     PAGE;
2480     POKE (CAPSLOCK, ON);
2490     FINISHED := FALSE;
2500     WRITELN;
2510     WRITELN;
2520     WRITELN ('MENU');
2530     WRITELN;
2540     WRITELN ('SELECT ONE OF THE FOLLOWING');
2550     WRITELN ('OPTIONS ');
2560     WRITELN;
2570     WRITELN;
2580     WRITELN ('1) READ THE INTRODUCTION');
2590     WRITELN ('2) CREATE DATA');
2600     WRITELN ('3) READ DATA');
2610     WRITELN ('4) SAVE DATE TO DISK');
2620     WRITELN ('5) LOAD DATA FROM DISK');
2630     WRITELN ('6) EXIT THE PROGAM');
2640     WRITELN;
2650     WRITELN;
2660     WRITE ('MAKE YOUR SELECTION ');
2670     READLN;
2680     READ (ANS);
2690
2700
2710     CASE ANS OF
2720       '1' : SONGANDDANCE;
2730       '2' : FILLDIRECTORY;
2740       '3' : READDIRECTORY;
2750       '4' : DISKSAVE;
2760       '5' : DISKLOAD;
2770       '6' : FINISHED := TRUE
2780     END;
2790
2800
2810   UNTIL FINISHED = TRUE;
2820
2830
2840     {FINALE}
2850     PAGE;
2860     WRITELN;
2870     WRITELN;
2880     WRITELN ('END OF DEMONSTRATION');
2890     WRITELN;
2900     WRITELN
2910 END.
```

The Pascal disk handler described in this article and program works only within a compiled Pascal program and, regretfull , can not be used to save Pascal source code. The procedures DIN, DOUT, and GETNAME are completely modular. They may be copied directly from this program into the appropriate section of any Pascal program you care to write which requires access to LKDOS. The procedures DISKSAVE and DISKLOAD may also be used but remember to change the name within the parentheses of ADDR() and SIZE() to the name of the variable within your program that you wish to save or load. The ".C" extension required by data file saves within Basic when using LKDOS is not required by the Pascal disk handler, however, it may be good practice to use the extension ".P" to indicate that the data that has been saved is intended for a Pascal program rather than a Basic program. All the LKDOS error codes are operational and will stop your program without crashing providing that the compiled code is accessed through a Basic loader program, that a PRINT USR is used rather than RANDOMIZE USR and that there is at least one line of Basic after the USR call. Example:

```
  10 REM TYPICAL COMPILED PASCAL PROGRAM LOADER
  20 BORDER 0: PAPER 0: INK 7: CLS
  30 PRINT #4: LOAD "MYPROG.C1" CODE 27000
  40 CLS: PRINT USR 27000
  50 STOP
9000 REM SAVE LOADER TO DISK
9010 PRINT #4: SAVE "LOADER.B1" LINE 1
```

On behalf of Larry and myself I hope that this Pascal disk handler will prove to be useful to all who wish to do serious programming in Pascal which requires disk access.

ZEBRA/TIMEX FDD DISK SYSTEM

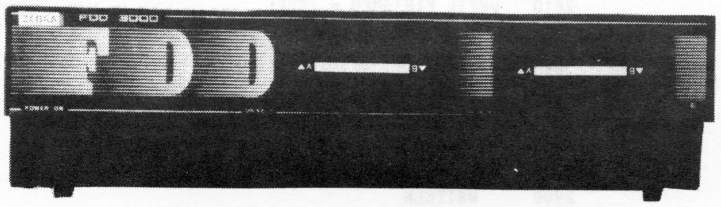
MACHINE CODE TRACK READER

by Mike Finn

In order to write more advanced programs for the Zebra FDD disk drive system, we need to know more about it's operating system. To write disk utilities we need to be able to see exactly what is on disk without a basic program using basic TOS (Timex Operating System) commands as intermediary. Since TOS is stored on the first four tracks of the disk and downloaded to controller RAM, we need disk reading and writing utilities to make any desired changes to TOS, or to develop utilities such as a program which would recover disk data after accidental erasure.

The following machine code program will read all the sectors of any given track on the disk and store the 4K bytes of disk track data in Home RAM at 7000 hex for access by a monitor disassembler machine code program. The track reader code resides at E000 to E050 hex. I have used both Zeus Monitor and HOT-Z AROS with it. In fact, I have used HOT-Z AROS in all 3 banks: home, dock, and exrom with 32K non-volatile memory board in the dock cartridge slot with this track reader program. The code can be readily modified to be used at some other location if you are using some other disassembler.

```
E000 00                          Storage space for drive #
E001 00                          Storage space for track #
E002 210070      LD HL, 7000     Beginning of home ram
                                 storage for disk track
                                 contents
E005 3A01E0      LD A,(E001)
E008 57          LD D,A          D contains track #
E009 3A00E0      LD A,(E000)
E00C 4F          LD C,A          C contains drive unit #
E00D FDE5        PUSH IY         Page in ZEBRA interface
E00F FD210000    LD IY,0000          rom/ram
E013 CD0800      CALL 0008
E016 FDE1        POP IY
E018 1E00        LD E,00         E will hold sector #
E01A 3E1B        LD A,1B         Disk command to read sector
                                 contained in E of track in
                                 D of drive in C
E01C 320021      LD (2100),A     Command buffer
E01F C5          PUSH BC         Save request parameters
E020 D5          PUSH DE
E021 E5          PUSH HL         Save current home ram
                                 download address
E022 CD0806      CALL 0608       Calls command sending
                                 routine
E025 CD2606      CALL 0626       This routine controls for
                                 reply from disk
E028 3A0221      LD A,(2102)     This system variable holds
```



```
E02B A7          AND A          TOS error code
E02C 0600        LD B,00        Test for error.
                                If an error occurred return
E02E 4F          LD C,A             to basic with error
E02F 201C        JR NZ,E04D         reason in BC
E031 E1          POP HL         Restore home ram address and
E032 110020      LD DE,2000         download from ZEBRA ram
E035 EB          EX DE,HL           buffer to home
E036 010001      LD BC,0100
E039 EDB0        LDIR
E03B EB          EX DE,HL
E03C D1          POP DE         Restore sector/track
                                parameter
E03D 7B          LD A,E         Pick up next data sector of
E03E C607        ADD A,07           track
E040 E60F        AND 0F
E042 5F          LD E,A
E043 C1          POP BC         Restore drive parameter
E044 20D4        JR NZ,E01A     If all 16 sectors not read
                                then loop back for next one
E046 010000      LD BC,0000     If all 16 read set error
                                report code to 0
E049 CD0306      CALL 0603      Page out ZEBRA rom/ram
E04C C9          RET            Return to basic to call up
                                either monitor or error
                                message
E04D E1          POP HL         Clear stack and return to
E04E E1          POP HL         basic
E04F E1          POP HL
E050 18F7        JR E049
```

Relative addressing is used as much as possible within the machine code. The only absolute address calls are to a jump table in Zebra ROM, as described in Appendix F of the user manual, so all versions of the FDD should run this code. The only lines that need to be changed to relocate this code for use with another monitor/disassembler are lines 3, 4, and 6. You will need to change the absolute addresses used in those lines to match your memory layout.

A basic program is used to load the code from disk, poke the track and driver parameters, load your favorite monitor, run the code, and enter the monitor program to view the disk data. In the basic program you must replace lines 30 to 38 and 140 to 148 with your own monitor/disassembler loads and calls. I use one of the following depending on my system configuration.

ZEUS monitor:

```
30 LOAD * "ZEUSMON.COD"CODE
140 IF BC=0 THEN PRINT USR 62137
```

HOT-Z AROS:

Home:

```
30 LOAD * "HOT-Z2,5.COD"CODE
140 IF BC=0 THEN RAND USR 32776
```

Dock:

```
140 IF BC=0 THEN OUT 255,0: OUT 244,
    240: RAND USR 32776
```

Exrom:

```
140 IF BC=0 THEN OUT 255,128: OUT 24
    4,240: RAND USR 32776
```

Mostly I keep HOT-Z in Exrom. There are several "bugs" in the FDD's initiation routines when dock bank cartridges are present. I avoid these conflicts by keeping HOT-Z AROS in exrom from 8000 to DFFF hex.

Once you enter the monitor you shouldn't need to return to basic. Parameters at E000 and E001 hex can be poked using utilities in the monitor program and both ZEUS and HOT-Z have code execution routines. For ZEUS, I use the DISASSEMBLE command to read code and the EDIT command to read data. The EDIT command is also used to change parameter values at E000 and E001. I use the GOTO command to rerun the code at E002. For HOT-Z, the normal read mode can disassemble code and the display switch (SS-G) will read data. I can enter the edit mode (SS-A) to input new parameter values and use the run command (CSS-RUN) to rerun the track reader code. HOT-Z has an advantage over ZEUS for reading files containing basic programs. ZEUS won't display the characters corresponding to codes over 127.

For those whose monitors do not include an assembler, the following loader may be used to enter the code.

```
  1 REM ZEBRA Disk Drive
        Track Reader Utility
        by Mike Finn
  2 REM This utility requires a
        monitor/disassembler
        machine code program
  3 REM Program allows for
        monitor code to occupy
        addresses from 8000
        to DFFF Hex or from
        E051 to FF57 hex
  4 REM E000 to E050 is reserv-
        ed for track reader m/c
  5 REM 7000 to 7FFF is reserv-
        ed for disk data
 10 CLEAR 28671
 20 LOAD *"TRACKRDR.COD"CODE
 30 REM ***********************
 32 REM
 34 REM Replace this REM with
        a LOAD * instruction to
        load your favorite mon-
        itor disassembler from
        Disk
 36 REM
 38 REM ***********************
 40 PRINT 'TAB 5;"DISK REVIEWIN
G UTILITY"
 50 INPUT "Source Drive (A TO D
) ";D$
 60 LET D=CODE D$
 70 IF D>=97 AND D<=100 THEN  L
ET D=D-32
 80 LET D=D-65: IF D<0 OR D>3 T
HEN  GO TO 50
 90 POKE 57344,D
100 INPUT "Track (0 to 39) ";T
110 IF T<0 OR T>39 THEN  GO TO
100
120 POKE 57345,T
130 LET BC=USR 57346
140 REM ***********************
142 REM
144 REM Replace this REM with
        IF BC=0 THEN RAND USR
        ( monitor address )
146 REM
148 REM ***********************
150 PRINT "ERROR ";BC;"! Please
see user manual."
```

```
 10 REM Machine code loader
 15 RESTORE
 20 FOR I= 57344 TO 57425
 25 READ A: POKE I,A: NEXT I
 30 DATA 000,000,033,000,112,05
8,001,224,087,058,000,224,079
 35 DATA 253,229,253,033,000,00
0,205,008,000,253,225,030,000
 40 DATA 062,027,050,000,033,19
7,213,229,205,008,006,205,038
 45 DATA 006,058,002,033,167,00
6,000,079,032,028,225,017,000
 50 DATA 032,235,001,000,001,23
7,176,235,209,123,198,007,230
 55 DATA 015,095,193,032,212,00
1,000,000,205,003,006,201,225
 60 DATA 225,225,024,247
```

"S" AND "Q" KEYS WITH "CAT*" COMMAND

by Mike Finn

After reading Ronald Havlen's FDD Express (Oct. '87) newsletter about the problem with stopping the scrolling on the CAT* command with keys "S" and "Q", I began to experiment and I soon discovered that these keys will work occasionally. Here is a small program which shows that an even larger problem exists.

First, make sure you have a disk with a large enough directory to require screen scrolling. One fast way to do this is to run the following program:

```
10 FOR I = 1 TO 40
20 LET A$ = STR$ I
30 DIM* A$
40 NEXT I
```

Now delete that program and enter the following:

```
10 LET I = 0
20 CAT*
30 LET I = I + 1
40 GOTO 20
```

Run this program and while it is running, press and hold the "S" key. Be patient, as this may take a minute or so. Eventually, the scrolling will freeze and will remain this way until the "Q" key is pressed. Press the "Q" key and the screen will begin scrolling again. Immediately press CAPS SHIFT and BREAK. Enter as a direct command PRINT I. The first time I tried this it took 33 repetitions of the loop from lines 20 to 40 before the scrolling stopped. Other times it took as few as four loops.

This shows that the "S" and "Q" keys do work sometimes...the problem is why don't they work all of the time.

I've also noticed another problem which may be similar to the systems non-response to the "S" key. Let's clean up our disk's directory with the following program.

```
10 FOR I = 1 to 40
20 LET A$ = STR$ I
30 ERASE A$
40 NEXT I
```

When you run this program, TOS (Timex Operating System) will ask you to confirm that you want to erase each directory entry. Answer "Y" each time and take notice of how long a wait there is between when you press "Y" and when TOS recognizes that you replied. The first six times I answered "Y", TOS recognized my input immediately, but the following two times, there was noticeable lag. on one occasion, I tapped the "Y" key several times before the system finally responded. I have not had this problem any other time, so I assume the problem is not a defective keyboard. (I would like to know if these things I've written about are peculiar only to my setup or if they are universal among all Zebra FDD owners.)

I have been working on a disassembly of the Zebra interface ROM. I believe the I've identified the subroutine that deals with the use of the "Q" and "S". I am still investigating this routine and have nothing final to report, but on my first glance, I see no reason why it shouldn't work consistently.

My initial review shows that when the Zebra interface ROM sets up a TOS command to the disk controller, it then controls a response using a subroutine which the Zebra Disk Drive Technical Manual (page 28) calls RESPOSTA. This routine lies between 0688 hex and 06DB. If TOS wants a write out of text in the data buffer (2000 to 20FF hex) or from the error message section of the command buffer (210D to 212D), the routine at 03EB to 0423 is called. It is this routine which contains the "S" and "Q" key tests. One noteworthy point about this routine is that it only tests the "S" key at the beginning of the screen printout of text. It can print 256 bytes from the data buffer before it has to return to the original calling routine. If a directory printout contains less than 257 bytes, the "S" key will not stop it in the middle of a transfer since it is only tested prior to the data printout.

Can a long directory be contained in just 256 bytes? If you look at the line by line printout of the previous sample directory, you will notice that it is mostly empty spaces. The TAB function can easily generate all those spaces in just a few bytes of machine code, so pressing "S" may not necessarily stop even a large directory from scrolling.

Zebra ROM subroutines called by the "S" and "Q" key routine:

```
000A FDCB01AE RES 5,(IY+01)   RESET SYSTEM VARIABLE FLAG WHICH
                              TRACKS KEYHITS
000E FDCB01DE SET 3,(IY+01)   SET CURSOR MODE L
0012 CD3003   CALL 0330       CALL KEYBOARD SCANNER IN
0015 B002     DEFB 020B         HOME ROM
0017 AF       XOR A           CLEAR FLAG REGISTER
0018 FDCB016E BIT 5,(IY+01)   TEST WHETHER KEYHIT FOUND
001C C8       RET Z           IF NOT, RETURN TO CALLING ROUTINE
001D 3A085C   LD A,(5C08)     SYSTEM VARIABLE - LAST KEY
0020 FE61     CP 61           TEST FOR UPPER CASE
0022 D8       RET C           RETURN IF UPPER CASE
0023 E6DF     AND DF          IF LOWER CASE, CONVERT TO
0025 C9       RET               UPPER CASE AND RETURN
0026 3E0D     LD A,0D         CARRIAGE RETURN
0028 CD3003   CALL 0330       CALL RST 10 IN HOME ROM
002B 1000     DEFB 0010         TO TRANSMIT CARRIAGE RETURN
002D C9       RET
```


DID YOU MISS THE FAIR?

If so, you'll be glad to know that you too can get in on the specials that were offered! We want to send a list of our special "AFTER THE FAIR" sale items so that you can get in on the savings! All you do is send a legal sized SASE and we'll send it out to you with all due haste! AND...If you would like to order a souvenier packet of specials and a program from the show, just send a check or MO for $3 and we'll send it out! (Packet includes specials from most attending vendors.)

For RMG's BIG 70+ page catalog, send $3, refunded first order.

RMG ENTERPRISES
1419 1/2 7TH STREET
OREGON CITY, OREGON 97045
503/655-7484 * NOON-10 TUE-SAT

printout routine:

```
03EB E5      PUSH HL      SAVE HL, ADDRESS OF TEXT TO BE
                            PRINTED OUT
03EC 3E02    LD A,02      OPEN CHANNEL # 2; 0330 IS THE
03EE CD3003  CALL 0330      CBAS ROUTINE USED TO CALL HOME
03F1 3012    DEFB 1230      ROM ROUTINES
03F3 3EFF    LD A,FF      POKE SYSTEM VARIABLE WITH FF FOR
03F5 328C5C  LD (5C8C),A    CONTINUOUS SCROLL OF SCREEN
03F8 CD0A00  CALL 000A    KEYBOARD SCANNER ROUTINE
03FB FE53    CP 53        IS IT THE "S" KEY?
03FD 2007    JR NZ,0406   IF NOT,CONTINUE WITH PRINTOUT
03FF CD0A00  CALL 000A    IF IT IS "S", KEEP SCANNING
0402 FE51    CP 51        IS IT THE "Q" KEY?
0404 20F9    JR NZ,03FF   IF NOT, THEN KEEP SCANNING
                            KEYBOARD UNTIL "Q" IS PRESSED
0406 CD2600  CALL 0026    THIS DOES HOME ROM RST 10 TO
                            TRANSMIT A CARRIAGE RETURN
0409 E1      POP HL       RESTORE POINTER TO BUFFER BYTES
040A E5      PUSH HL
040B 7C      LD A,H       THIS TESTS WHETHER HL POINTS TO
                            COMMAND BUFFER 210D HEX OR TO
                            DATA BUFFER 2000 HEX
040C 1F      RRA          CARRY WILL BE SET FOR COMMAND
                            BUFFER, AND RESET FOR DATA
                            BUFFER READOUTS
040D 0600    LD B,00      THIS SETS UP A PRINTOUT OF
                            UP TO 32 BYTES, THE MAXIMUM
                            SIZE OF THE MESSAGE AREA IN THE
                            COMMAND BUFFER
040F 3802    JR C,0413
0411 0621    LD B,21      THIS SETS UP A PRINTOUT OF UP TO
                            256 BYTES, THE MAXIMUM LENGTH OF
                            THE DATA BUFFER, 2000 TO 20FF
0413 E1      POP HL       RESTORE BUFFER TEXT ADDRESS
0414 04      INC B        TESTS THAT WE DON'T EXCEED THE
0415 78      LD A,B         MAXIMUM SIZE OF BUFFER
0416 FE21    CP 21
0418 C8      RET Z        IF SO, RETURN TO CALLING ROUTINE
0419 7E      LD A,(HL)    PICK UP TEXT BYTE POINTED TO BY
                            HL
041A B7      OR A         TESTS TO SEE IF WE REACHED END
                            OF DATA MARKER, 00 HEX
041B C8      RET Z        IF SO, RETURN TO CALLING ROUTINE
041C 23      INC HL       IF NOT END OF DATA OR END OF
041D E5      PUSH HL        BUFFER THEN GET NEXT ADDRESS TO
                            BE PRINTED OUT AND SAVE IT
041E CD3003  CALL 0330    USE CBAS TO RUN HOME ROM RST 10
0421 1000    DEFB 0010      FOR TEXT BYTE IN THE A REGISTER
0423 18EE    JR 0413      REPEAT THE END OF DATA AND END OF
                            BUFFER TESTS
```

Note that the keyboard is only tested once for the "S" key, then up to 265 bytes are printed out. When this is completed we return to REPOSTA which sends a DONE message to TOS. If TOS has anything else to output to the screen, it repeats the request for data printout and up to 256 bytes can again be printed. So we only get the chance to stop the screen scrolling every 256 bytes.

When you try the program, you will see the scrolling freeze occasionally and will have to press "Q" to restart it. This shows that the routine does work. I see no software bugs to prevent it from working all the time (i.e., every 256 bytes). I don't know enough about the hardware mechanisms involved in paging in and out the Zebra FDD shadow ROM, but I suspect a timing problem or a keyboard debouncing/reading problem more than a software problem. Does anyone have any suggestions for further study so we can nail down the source of this problem?

OLIGER SAFE DISK SYSTEM

USING OLIGER SAFE DOS VERSION 2.52

by Dick Wagner

The final OLIGER SAFE DOS (Disk Operating System) on EPROM is now available as version 2.52. The system is more than just a disk operating system. The additional utilities that John Oliger provides are interesting and useful.

Here are some of the latest features:

1. A fast FOR--NEXT loop routine
2. ERASE /"Filename" command
3. improved cataloging (CAT command)
4. RESTORE /"New disk name" command
5. MERGE /"Name" command

Along with these, there is a well-coordinated version of the MSCRIPT word processor available to use with SAFE DOS, and it is now possible to operate two disk drive operating systems at will, without changing disks! (i.e., OLIGER and LARKEN)

The fast FOR--NEXT loop utility provides a constant speed whenever it is used in a program, giving 9 to 50 times faster operation. Only one such loop is permitted, but it may be mixed with regular BASIC loops. It is simple to implement. The variable must be assigned at the beginning of a program, such as:

```
  5 LET /k=1
200 FOR /1 TO 100
255 NEXT
```

The ERASE /"Filename" command permits erasing any disk file, which is great for cleaning up a disk. All consecutive files following the erased file are moved up and the catalog is corrected without blank lines.

The new CAT extended command produces an improved screen display with an added column which shows the starting address for code and data files.

The extended command FORMAT /"name" has been in use from the beginning, as it is always necessary to give the disk a name (even if the name is only " ").

Now the user can change disk names at will with the RESTORE /"New disk name" command. This is handy for formatting disks in advance.

With the new MERGE /"name" command, Oliger makes it possible to append a program to an existing program, without seriously polluting the current program!

Printing a hard copy of the disk catalog is a snap. In the immediate mode, type: LET /P=0 and OPEN #2,"p". Now type CAT, and the display is shunted to the printer in place of the screen.

The OLIGER SAFE disk system and hardware has the unique ability to be compatible with the LARKEN DOS (Disk Operating System), which is supplied in cartridge form, and is available from LARKEN (and RMG Enterprises). Now the user can operate both systems interchangeably at will (with the LARKEN disk in drive 0 and the OLIGER disk in drive 1...for example). The extended commands can be sent to either disk, and even some OLIGER commands can be used in LARKEN programs, such as the fast FOR--NEXT loop routine. As I prefer using the OLIGER DOS, I can purchase programs available only for the LARKEN DOS, make the appropriate program changes, and save a version on the OLIGER.

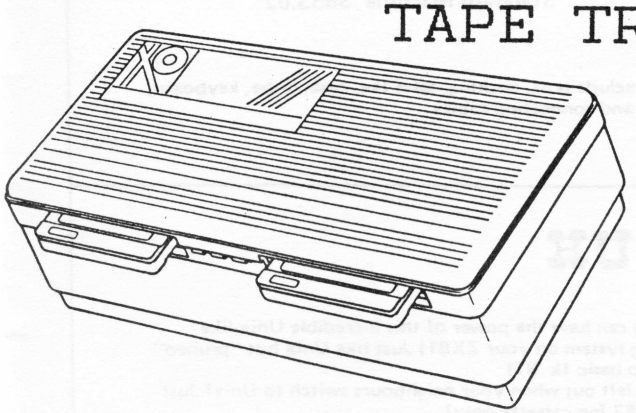
My personal favorite (and much used) disk operation is in conjunction with MSCRIPT version 5.3. I make a MSCRIPT utility save on each disk I use with this program. A special FILE 0 program is used with LOAD to display the catalog with a moveable cursor. Select MSCRIPT, press ENTER and there it is. Issue the CAT command in the MSCRIPT menu and the catalog is displayed. Select a program to LOAD into MSCRIPT as text, and there is the complete text, including a list of printer commands used with that particular text. A quick delete of the text leaves the printer codes to use as reference.

For convenience, I also have the printer codes (52 possible) stored on the same disk as text. It can be loaded at any time, referred to, codes added to the list already in the reference lines, and then be deleted. The alternative is a printed listt.

Many large printers provide graphic capabilities where every print pixel is defined. John Oliger uses the OUT 127,n direct port approach for this type of programming. I prefer to use the LPRINT method commonly used in many computer programs. The LET /P=0 command will not properly send printer codes greater than 127. The word from John, is to make 2 pokes: 23300,60 and 23301,3 to overcome this problem, so that LPRINT CHR$ n will work properly.

ROTRONICS WAFADRIVE

TAPE TRANSFER UTILITY



The following utility program was supplied to TIME DESIGNS by Dave Maccarone, a former authorized distributor of the Rotronics Wafadrive. Rotronics is not longer in business. It should be noted that the following "Tape Transfer Utility" is for the Spectrum computer or the Spectrum-Emulated Timex Sinclair 2068.

When transferring software from cassette to wafer, some kind of transfer utility program is virtually indispensable. The program given here is written largely in BASIC and sets up its own machine code subroutines. The three facilities provided by the program are:

1. READ HEADER: Essential for analysing the attributes of files on cassette. Displays the auto-run line number of BASIC programs, location and length of code files. It works by reading the program header which preceeds the file on tape. Beware of false headers!

2, RECLAIM RAM: This option effectively de-initializes the Wafaderive Operating System and reclaims all RAM used. The BASIC program is pre-served.

3. STOP: This can be used to load a BASIC program such that it is prevented from auto-running once loaded. It will thus cope with "anti-merge" programs. The auto-run line number is displayed on screen with the program name. The utility program itself is overwritten.

The machine code subroutines used by the program are all relocateable and can be lifted for use in your own programs. Memory saving techniques have been used in the BASIC to assist when working in "con-fined spaces". The program can be further reduced in size if required by splitting it into three parts and/or by loading the machine code straight into the printer buffer as a separate file.

```
  10 PAPER VAL "5": INK NOT PI: BORDER VAL "5": GO SUB VAL "1e3"
  20 CLS : PRINT "Enter number:"'''
  30 PRINT " 1 READ HEADER"'"   - loads header from cassette
& displays file attributes."
  40 PRINT ''" 2 RECLAIM RAM"'"   - de-initialises WOS &
  reclaims RAM."
  50 PRINT ''" 3 STOP"'"   - loads BASIC program from      casse
tte, preventing auto-      run."
  60 LET a$=INKEY$: IF a$<"1" OR a$>"3" THEN GO TO 60
  70 CLS : GO TO VAL "100"*VAL a$
 100 REM  HEADER READ
 110 PRINT " HEADER READ "'''"Play tape...": RANDOMIZE USR VAL "2
3320": CLS : LET a=VAL "23296"
 120 LET t=PEEK a: LET a=a+SGN PI
 130 PRINT "TYPE:","Program" AND (NOT t);"Number array" AND (t=S
GN PI);"String array" AND (t=VAL "2");"Bytes" AND (t=INT PI)
 140 PRINT '"NAME:",: FOR n=SGN PI TO VAL "10": PRINT CHR$ (PEEK
 a);: LET a=a+SGN PI: NEXT n
 150 PRINT ''"LENGTH:",PEEK a+VAL "256*PEEK (a+1)": LET a=a+VAL
"2"
 160 LET b=PEEK a+VAL "256*PEEK (a+1)": IF NOT t THEN PRINT '"AU
TO-RUN:",: IF B<VAL "1e4" THEN PRINT b
 170 IF t=INT PI THEN PRINT '"START:",b
 180 PRINT £NOT PI;"Press a key to return to menu": PAUSE NOT PI
: GO TO VAL "20"
 200 REM  RECLAIM
 210 PRINT " RECLAIM RAM "
 220 GO SUB VAL "2e3": CLEAR #: RANDOMIZE USR VAL "23340": GO TO
 VAL "20"
 300 REM  STOP
 310 PRINT " STOP "''"WARNING! - this program is lost when selec
ted.": GO SUB VAL "2e3"
 320 PRINT "The number shown on screen is  the line from which
the program would normally auto-run."''"Play tape..."'';USR VAL
 "23369": STOP
1000 RESTORE
1010 DATA "221","33","0","91","221","229","17","17","0","175","5
5","205","86","5","221","225","48","242","251","201"
1020 DATA "219","12","205","46","10","205","159","0","42","99","
92","34","101","92","42","146","92","34","104","92","33","182","
92","1","244","8","195","232","25"
1030 DATA "1","34","0","247","213","221","225","221","54","1","2
55","221","54","15","128","62","1","50","116","92","205","58","7
","62","255","50","68","92","237","75","66","92","205","27","26"
,"207","255"
1040 FOR a=VAL "23320" TO VAL "23405": READ a$: POKE a,VAL a$: N
EXT a: RETURN
2000 REM  CONTINUE?
2010 PRINT '''Press Y to proceed or any other key to return to m
enu"
2020 PAUSE NOT PI: LET a=PEEK VAL "23560": IF a<>CODE "Y" AND a<
>CODE "y" THEN RUN
2030 CLS : RETURN
```

****

P U Z Z L E O F T H E M O N T H

****

CEDRIC R. BASTIAANS

The GIVENS of this puzzle are:

A. There are 5 children, all more than 1 year old and younger than 25.
B. Their combined ages total 40.
C. The product of the boys' ages is 39 times the product of the girls' ages.
D. Next year, the product of the boys' ages is an even number of times the product of the girls' ages.

CAN YOU COME UP WITH A PROGRAM FOR OUR COMPUTERS TO FIGURE OUT THE AGES OF THE FIVE CHILDREN???

Note: This is NOT a real ad. Please do not send real money. Thanks to William Scott for contributing this humorous little ditty from Australia.

Exclusive advance purchase !

SINCLAIR ZX89

We've done it again! Be the first in your street to own Sinclair's new baby, the incredible ZX89! Just look at these features:

- Powerful, Cray 1 compatible processor on one chip!
- 14 Megabytes of RAM on one chip!!
- Advanced touch-sensitive multifunction keyboard – each key has 27 functions!!!
- Twin 10 Mbyte winchester disks – new, ultra-mini design exclusive to ZX89!!!!
- Plugs into your own TV, runs for a year on two micro batteries (not supplied)!!!!!
- Auto repeat function on exclamation mark key !!!!!!

All this for just $35.00 inc sales tax, post, packing, all connecting cables, and all the software you can get into your house! (Send cash only – no cheques, credit cards, POs, Lan Choo Labels, etc accepted. Strictly mail order only.)

Microbee Pacemaker

That's right! Now you can control your *own heartbeat* with this powerful software package. All connectors plus scalpel blades supplied – install it yourself with our easy-to-follow instructions. Lifetime guarantee. Supplied on cassette for just $15.95 inc p&p (no refunds). Ask about our very own life insurance scheme exclusive to Microbee Pacemaker owners!

Games software

Dam: Build your own Franklin below Gordon obstruction, fend off the Greeny attackers by puncturing their rubber dinghy. With full colour, sound, fast-moving graphics. $20.00

Orphan attack: Defend the food-laden table from invading hordes of hungry orphans. Can you eat it all before they do? $16.00

Hitch-hiker: You're driving along a dark road. Suddenly a hitch-hiker appears in the gloom. Can you run him over before he jumps clear? $520.00

The Stones of HI'Fydgra: Ultimate 'Adventure' type game, lasts for weeks, no-body has yet managed to win – will *you* be the first? Explore the caverns of Ygnbs, the Magician of Ggon; fight Typo the mystic Gzknw and his followers, the Pfinds of Tzdi. Find the Treasure of Pjjjj and you've won! $2.00 inc free dictionary.

Zap the Zits: Can you stop those pesky zits from spreading? The faster you zap them, the faster they come back! Find the secret 'Clearasil' button to win. Full sound and colour. $45.00

Buy yours NOW!

DON'T WAIT TILL NEXT MONTH FOR THE VZ200

Beat the delay – buy your VZ200 Dick Smith color computer now from us *over the counter!* How do we do it? Simple! As soon as we heard of the VZ we bribed security guards at the Hong Kong factory to shoot machines out the side door!!

Standard Machine $853.02

Prices exclude post, packing, sales tax, case, bribe, keyboard, manual and connecting cables.

UNUX

Now you can have the power of this incredible Unix-like operating system on your ZX81! Just like Unix but "pruned" to fit into basic 1k '81!
Don't be left out when your neighbours switch to Unix! Just $23,427.63 (on cassette only).

Business software

MoralStar: New addition to your word processing suite. Automatically searches your text for dirty words, expletives, *double-entendres*, etc, and replaces them with 'clean' text which even F Nile would approve. A 'must' for TV scriptwriters! $750.00

TLO Toolkit: Trying to get to grips with The Last One? Our TLO Toolkit makes the job a cinch, gets TLO working, produces exactly the code you want. *Definitely* the last program you'll need to buy (revised version coming in September). $540.00

VIC-370: Give your VIC 20 the power of an IBM 370 with this super conversion program! You too can have mainframe computing power at your fingertips in the comfort and privacy of your own home. Requires 1,432 16k RAM packs. On cassette only $37.50

Innuendo: Not a game but an indispensable addition to your word processing system. Scans your text and inserts dirty words, expletives, *double-entendres*, etc, in place of 'clean' words. A 'must' for TV scriptwriters! $23.95

'Bumper' Bundle: All your favourite packages on one disk! Wordstar, The Last One, dBase II, Visicalc, Unix, MBasic and seven versions of Pascal, all available for the first time on a single disk! $9.20 (discounts for bulk orders and educational institutions). (Manuals not available at time of going to press.)

The Editor: Incredible new 'artificial intelligence' program replaces magazine editors with a computer. You too can produce your very own micro magazine in the comfort and privacy of your own home! As used by J Pierce, L Belle and several others. $2500.19.

WILD BILL'S COMPUTER RODEO

PO BOX 835539A SYDNEY

(02) 747 6311

THE SOLUTION OF THE PUZZLE OF THE MONTH

CEDRIC R. BASTIAANS

First, we'll give the children some simple names, very simple like a, b, c, d and e.
Then, we will assume that there are 3 girls and 2 boys. Maybe it's the other way around, but we'll see.
Furthermore, the number 39 can only be composed of 1x39 or 3x13. What this means is that one of the boys simply HAS to be 13 years old, while another boy's age HAS to be divisible by 3. We may thus write:

a=13 and
b=3t, where t=1,2,3....8 (maximum 8, because no one can be older than 24).

The girls being called c,d and e, we may write (algebraically):

39cde = 13(3t) or cde = t .....(1)

Also:

c+d+e+13+(3t) = 40 or c+d+e = 27-3t .....(2)

Since c,d and e are older than or as old as 2 years, we can write:

cde >= 8 .....(3)

and c+d+e >= 6 .....(4)

From equations (1) and (3) follows that:

t >= 8 .....(5)

On the other hand, from (2) and (4) we find:

t <= 7 .....(6)

Equations (5) and (6) contradict and the truth is therefore that we have THREE boys and only TWO girls!

The situation for the 3 boys a, b and c and the 2 girls d and e is therefore a trifle different:

39de = 13(3t)c or de = tc or c = de/t .....(7)

while, however, equation (2) still holds true!

Substituting for c in (2) yields:

3t+de/t+d+e = 27
or $3t^2$+de+dt+et = 27t
or t^2+de+dt+et = 27t-$2t^2$
or (d+t)(e+t) = t(27-2t) .....(8)

We may thusly write the following program:

```
10 LET A=13
20 FOR T=2 TO 7
30 LET B=3*T
40 REM EQUATION (8): LET SUM=T*(27-2*T)
50 FOR D=2 TO 24
60 FOR E=D TO 24
70 REM EQUATION (7): LET C=D*E/T
80 IF C=INT(C) AND (D+T)*(E+T)=SUM THEN
   GOSUB 1000
90 NEXT E
100 NEXT D
110 NEXT T
120 STOP

1000 REM NEXT YEAR AND SOLUTION
1010 LET X=(A+1)*(B+1)*(C+1)
1020 LET Y=(D+1)*(E+1)
1030 IF X/Y=INT(X/Y) THEN PRINT
     "THE BOYS ARE ";A;",  ";B;
     "  AND";C;" YEARS, THE GIRLS
     ARE ";D;" AND ";E;"."
1040 RETURN
```

With this program keyed into my TS2068, it displayed the solution in 11 seconds
BOYS 13, 9 and 8; GIRLS 4 and 6.

A Letter From Fred Nachbaur (concerning the PC8300 "Timex Clone")

Dear Tim,

I read with interest Bruce C. Taylor's article on the PC8300, since I have been quite deeply involved with this machine. I would like to clarify some of the points brought up by Mr. Taylor.

The 50/60 Hz. signal diode serves exactly the same function as pin 22 of the ZX81/TS1000 ULA. The video frame rate has nothing to do with the power-line frequency; instead, it is determined by the that unique blend of hardware and software that constitutes the ZX display system. In the PC8300, the 50/60 Hz. diode goes to an input port. Each time through the display loop, the software checks whether this port is high or low, adjusting the MARGIN system variable accordingly.

It should be noted that the machine (and the ZX81, for that matter) runs about 50% faster, overall, in SLOW mode, if it is in the 50 Hz. mode. This is because the system has more time between frames to work on your program. My ROM improvement (more about that later) allows MARGIN to be changed by the user, and always defaults on power-up at 60 Hz. regardless of whether the diode is connected or not.

There is most certainly a clock component on the PC8300 board. It is exactly the same as on a ZX81, a 6.5 mHz. ceramic element just to the left of the ULA, in front of the modulator. This behaves electrically just like a crystal; the only significant difference is that the frequency stability and precision is less than that of a crystal. For most jobs, however, the variance is negligible.

There is also most certainly a Z80A on the board. This is the large chip in the centre. Mr. Taylor may have been confused by the fact that some manufacturers of the Z80A give it their own part number. The designation 780C is particularly common. There is absolutely no difference between such chips and Z80A's marked as such.

Regarding place of origin, it does say "Made in Hong Kong." However, the manual is written in the Chinese dialect of the mainland. Draw your own conclusions.

There is nothing strange about the fact that the 1 REM location is 793 bytes higher on the PC8300, than it is on the ZX81 family. There is also nothing mystical about memory allocation. The only difference is the location of the display file. On the ZX81, it rides above the BASIC program. It therefore moves about as BASIC lines are added and deleted. On the PC8300, however, the display file is always at a fixed location BELOW the program area. What do you get when you multiply 24 lines by 33 bytes per line and add 1 for end-of-file marker? That's right, 793.

The PC8300 will successfully load ZX81 programs that are entirely in BASIC. It does this by looking at the VERS variable, adjusting how it perceives the incoming data as required. However, because of the way the display file location was modified, it will NOT load any BASIC variables associated with the Sinclair program. So if your program is, for instance, a mailing-list, you can load the program but not your data.

Worse, the different location of the first program line means that machine-code won't run unless it was written to be position-independent, and you adjust all USR calls accordingly. Even worse, no ROM calls are allowed since the ROM routines were completely scrambled, presumably to help prevent copyright hassles. For all practical purposes, one can consider that it is completely incompatible with ZX81 machine-code.

There are also some relatively minor, though potentially troublesome changes in the system variables. You have to be aware of these if you intend to write machine-code for this computer.

It is not possible to use a straight Timex or Sinclair ROM with the PC8300. Rather, it isn't practical. The reason is that the hardware of the display system is sufficiently different to cause the display to be "wonky," for lack of a better word.

I have developed a Timex-compatible ROM which runs all known low-res Timex programs, including machine-code. The only programs that won't run, are high-res programs and SOME "banner" programs.

The other hardware-dependent factor is the character set. These are NOT housed in the top 512 bytes of the ROM, as on the

Sinclair. Rather, they are contained within the custom chip, and are the reason that high-res won't work no matter what we do. There is no point, even, in trying to revector the I register; actually a blessing, since it is now available to the programmer. For instance, IM2 (interrupt mode 2) is theoretically usable with this machine. On the down side, some of the Sinclair punctuation (: ? and the pound sign) still remain game characters, and the grey graphics are right triangles and a "race car".

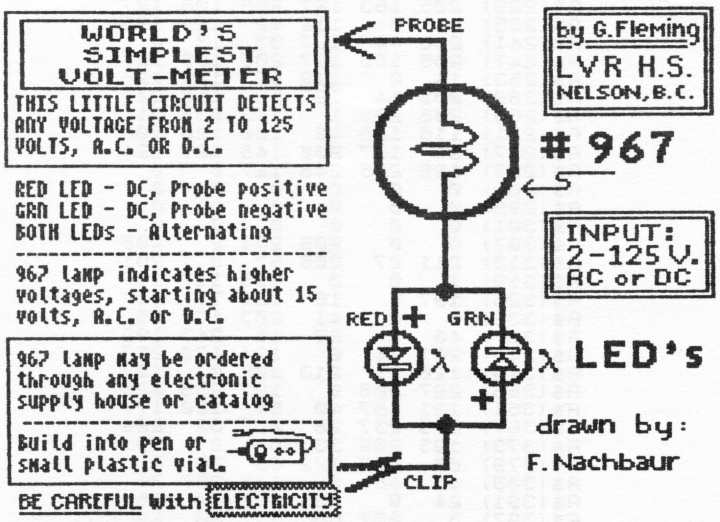
Otherwise, the PC8300 behaves just like a TS1000 with the new ROM. Even the keywords were reverted to their original key locations. Keywords are, of course, entered with a single key-stroke.
To make up for the few remaining shortcomings of my Timex-compatible ROM, it sports a few added features, some of which are not available either on the ZX81 or the stock PC8300. A new BEEP keyword turns beeping keys on or off. The REM command is now used to turn the blinking cursor on or off, in addition to its use as a REMark statement. A BASIC single-stepper ("debugger") is included. The machine can be set to make an audible noise when loading or saving a program.

Hardware-wise, there is nothing "different" about the electrical characteristics of the edge-connector lines. The problem is that some of them are missing. That's right, the edge traces go absolutely nowhere. Most of these (WAIT*, BUSRQ*, BUSAK*, HALT*, NMI*, etc.) are not commonly used by external peripherals. At least one, however, is relatively vital; this is M1*. If this line is brought out, the PC8300 will work with Timex/Sinclair-compatible 64K RAMpacks. Incidentally, no modification is needed to run machine-code in the 32-48K region. In this respect, it is like the TS1500.

Even without the M1* or other lines connected, the '8300 works with most ordinary Timex peripherals, such as the 2050 modem, 16K RAM, 2040 printer, many "big-printer" interfaces, and so on. One notable exception would be the Oliger Video Upgrade, again because the ROM does not contain the character patterns. Other devices that would not work include anything with an EPROM that overlays the ROM (e.g. the Memotech parallel interface), or contains ROM calls (e.g. the A&J Stringy-Floppy).

In my (admittedly biased) opinion, the PC8300 is virtually useless, unless its ROM is changed to make its claim of "Timex compatibility" at least 90% justifiable.

Regards,
Fred Nachbaur



UPDATE!
Take it to the limit
A Quarterly Magazine for the users
Supporting the Sinclair QL, Z88, and TS-2068
Subscription $15.00 Year. UPDATE Magazine,
1317 Stratford Ave., Panama City, FL 32404

CKTYPE 1000 (M/C VERSION)

by Earl V. Dunnington

Now thanks to Earl, we have a sum-checker program for the TS1000/TS1500/ZX81, Just like the one Stan Lemke introduced for the TS2068 in TDM, May/June '88.

The Machine Code is not relocatable and the program requires a minimum of 16K RAM. Those readers who wish to use the CKTYPE listings should LOAD the final BASIC version before proceeding.

Listing K is the final decimal machine code data, after deleting the unused portion of the Run Time Package. Included in the code is a routine to load a program from tape, defeating the autorun. This is necessary as some programs may contain machine code that would be loaded into the area where the code for CKTYPE 1000 will reside, if they were allowed to autorun. The loading routine will not work if you play the tape before the end of any program proceeding the one you wish to load.

Listing L is a program that will convert the machine code data into the corresponding character (CHR$) and enter it into the dimensioned string A$. This program will also produce a printout in the same format as Listing K, so that you can check the data. To correct an error, use the direct command:

LET A$(n)=CHR$ c

where n= the string character number and c= the correct decimal code.

WARNING: FROM THIS POINT ON, DO NOT USE CLEAR, DIM A$, NEW, OR RUN. These commands would erase A$.

Listing M is the CKTYPE printout for the program of Listing L.

After entering and checking all of the M/C data, program lines are to be deleted or added so that the program will now appear as in Listing N. After POKEing some additional machine code into the REM statement using the direct command GOTO 2 and deleting lines 2 to 6, this becomes the final CKTYPE 1000 M/C program. To SAVE the program on tape, use the direct command: GOTO 20.

Listing Ø is the CKTYPE for Listing N before using the command GOTO 2.

When LOADed, the program will:

Set RAMTOP to 32085

POKE the machine code stored in A$ into the addresses above RAMTOP starting with 32086

Clean the memory below RAMTOP in preparation for either typing in a program or LOADing a program from tape for which a CKTYPE listing is desired

Instructions for operating the programs will be displayed on the screen. Anyone needing more detailed instructions, send an S.A.S.E. to me at 4356 King Theodore Dr. Boynton Bch., FL 33436. Comments on this series of articles are also solicited.

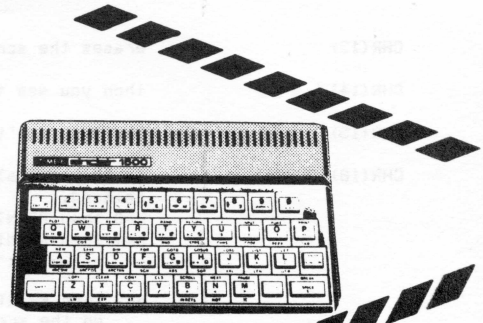
LISTING K

A$(1)	33	125	64	34	116	126
A$(7)	33	13	64	205	252	127
A$(13)	235	33	0	1	205	173
A$(19)	126	229	33	12	64	205
A$(25)	252	127	235	225	25	34
A$(31)	118	126	42	116	126	34
A$(37)	120	126	33	0	0	34
A$(43)	122	126	33	0	0	34
A$(49)	124	126	33	0	0	34
A$(55)	126	126	42	120	126	205
A$(61)	252	127	235	33	0	1
A$(67)	205	173	126	229	17	1
A$(73)	0	42	120	126	25	205
A$(79)	252	127	235	225	25	34
A$(85)	122	126	17	2	0	42
A$(91)	120	126	25	34	120	126
A$(97)	17	1	0	42	120	126
A$(103)	25	205	252	127	235	33
A$(109)	0	1	205	173	126	229
A$(115)	42	120	126	205	252	127
A$(121)	235	225	25	34	124	126
A$(127)	17	2	0	42	120	126
A$(133)	25	34	120	126	33	1
A$(139)	0	34	128	126	33	1
A$(145)	0	34	130	126	42	124
A$(151)	126	34	132	126	33	1
A$(157)	0	34	134	126	33	252
A$(163)	125	34	136	126	237	91
A$(169)	128	126	42	126	126	25
A$(175)	17	1	0	25	229	42
A$(181)	120	126	205	252	127	235
A$(187)	225	25	34	126	126	42
A$(193)	128	126	205	10	127	34
A$(199)	128	126	17	1	0	42
A$(205)	120	126	25	34	120	126
A$(211)	205	201	127	130	126	253
A$(217)	203	1	206	42	122	126
A$(223)	205	44	127	33	6	0
A$(229)	205	163	127	205	108	127
A$(235)	14	0	112	42	124	126
A$(241)	205	44	127	33	13	0
A$(247)	205	163	127	205	108	127
A$(253)	14	0	112	42	126	126
A$(259)	205	44	127	62	118	215
A$(265)	253	203	1	142	237	91
A$(271)	118	126	42	120	126	205
A$(277)	24	127	205	148	127	124
A$(283)	125	205	248	127	0	0
A$(289)	0	0	0	0	0	0
A$(295)	0	0	0	0	0	0
A$(301)	0	0	0	0	0	0
A$(307)	0	0	205	231	2	205
A$(313)	211	27	205	67	3	201
A$(319)	1	0	0	195	117	11
A$(325)	197	6	16	124	77	33
A$(331)	0	0	41	203	17	23
A$(337)	48	1	25	16	247	193
A$(343)	201	14	0	124	170	230
A$(349)	128	71	213	205	0	127
A$(355)	227	205	0	127	235	225
A$(361)	121	167	40	54	120	177
A$(367)	229	237	98	237	82	229
A$(373)	193	209	33	0	0	55
A$(379)	63	245	122	83	30	8
A$(385)	41	48	5	143	237	74
A$(391)	24	9	135	237	74	56
A$(397)	3	237	66	61	60	29
A$(403)	32	236	95	241	48	226
A$(409)	203	79	32	1	235	203

```
A$(415)  127  200   24   11  205  154
A$(421)  126  203   16  208   24    3
A$(427)  124    7  208  235  167  237
A$(433)   98  237   82  201  124  181
A$(439)   33    0    0  192   44  201
A$(445)    1    0  128   24    4  235
A$(451)    1    1  128  124  168  103
A$(457)  122  168   87  237   82   38
A$(463)    0  121  111  216  238    1
A$(469)  111  201   58   33   64  203
A$(475)   71   32   26  203   79   32
A$(481)   10  203  124   40    6   62
A$(487)   22  215  205    3  127    1
A$(493)  240  216   30  255  197  205
A$(499)  225    7  205  173   10  229
A$(505)  193   33   93   64   87   30
A$(511)   28  112  203  122   40    1
A$(517)  113  151  237  111  131  215
A$(523)  151  237  111  131  215  203
A$(529)  122  203  250   40  239  201
A$(535)  225  126   35  254  112   40
A$(541)   47  254  118   32    3  215
A$(547)   24  243  205  148  126   24
A$(553)  238  151   24    2   62  144
A$(559)   75   69   33   48   64   86
A$(565)  119  213  229  205  178   11
A$(571)  225  209  114  201  205   10
A$(577)  127  124  181  227   94   35
A$(583)   86   35  227  192  225  235
A$(589)  233  125  230   31   79  253
A$(595)  203    1   78   40   10  253
A$(601)  150   56  203  255  198   60
A$(607)  212  113    8  253  134   57
A$(613)  254   33   58   58   64  222
A$(619)    1  205  250    8  253  203
A$(625)    1  198  201  225   94   35
A$(631)   86   35  229  213   33    7
A$(637)    0   25    6    4   86   43
A$(643)   94   43  213   16  249  225
A$(649)  193  209  122   25  235  225
A$(655)  227  115   35  114  197  225
A$(661)    7   48    1  235  205   18
A$(667)  127  225  216  193  233  235
A$(673)  115  201  193  229  193  201
A$(679)  110   38    0  201
```

LISTING L

```
  20 DIM A$(682)
 170 PRINT "WORKING FROM LEFT TO
 RIGHT,     ENTER A DATA VALUE"
 180 FOR N=1 TO 682
 185 INPUT DATA
 190 LET A$(N)=CHR$ DATA
 200 NEXT N
 201 PRINT AT 0,0;"TO OBTAIN A P
RINTOUT OF A$,     TURN ON PRINT
ER AND PRESS A LET-TER KEY OTHER
WISE PRESS BREAK"
 202 PAUSE 32768
 203 FOR N=1 TO 682 STEP 6
 204 LPRINT TAB 0;"A$(";N;")";TA
B 8;CODE A$(N);TAB 12;CODE A$(N+
1);TAB 16;CODE A$(N+2);TAB 20;CO
DE A$(N+3);TAB 24;CODE A$(N+4);T
AB 28;CODE A$(N+5)
 205 NEXT N
 206 PRINT AT 0,0;"TO CORRECT AN
 ERRONEOUS VALUE,  USE THE DIREC
T COMMAND:          LET A$(N)=CHR
$ DATA              WHERE N=THE C
HARACTER NUMBER IN  THE STRING AN
D DATA IS THE COR-  RECT VALUE"
 210 STOP
```



LISTING M

```
 20    :  15  :  992
 170   :  54  :  2300
 180   :  21  :  1497
 185   :   6  :  539
 190   :  13  :  922
 200   :   3  :  417
 201   : 114  :  4667
 202   :  13  :  816
 203   :  29  :  2088
 204   : 163  :  8429
 205   :   3  :  417
 206   : 191  :  6918
 210   :   2  :  348
```

LISTING 0

```
1     :   6  :  483
2     :  21  :  1248
3     :  22  :  1249
4     :  22  :  1215
5     :  23  :  1335
6     :  23  :  1285
10    :   2  :  375
20    :  10  :  705
30    :   2  :  350
40    :  22  :  1236
50    :  23  :  1344
60    :  22  :  1581
70    :  21  :  1287
80    :  21  :  1485
90    :  23  :  1593
100   :  22  :  1218
110   :  23  :  1338
120   :  14  :  1031
130   :  10  :  717
140   : 190  :  9214
150   : 140  :  5818
160   :  91  :  3942
170   :  13  :  816
180   :  21  :  1497
190   :  22  :  1543
200   :   3  :  417
210   :   2  :  351
```

LISTING N

```
  1 REM 1234
  2 POKE 16510,0
  3 POKE 16514,49
  4 POKE 16515,81
  5 POKE 16516,125
  6 POKE 16517,201
 10 RETURN
 20 SAVE "CKTYPE"
 30 FAST
 40 POKE 16388,85
 50 POKE 16389,125
 60 POKE 32084,62
 70 POKE 32083,0
 80 POKE 32082,6
 90 POKE 32081,118
100 POKE 16386,81
110 POKE 16387,125
120 RAND USR 16514
130 GOSUB 10
140 PRINT AT 0,9;"CKTYPE 1000
";AT 2,0;"1. RAMTOP HAS BEEN SET
 AT 32085";AT 4,0;"2. PRESS A LE
TTER KEY TO MOVE   M/C ABOVE RAM
TOP AFTER MAKING A NOTE OF THE F
OLLOWING"
150 PRINT AT 8,0;"3. WHEN CURSO
R APPEARS, EITHER  TYPE IN A PRO
GRAM OR LOAD A PRO-GRAM FROM TAP
E USING THE DIRECT COMMAND: RAND
 USR 32394"
160 PRINT AT 13,0;"4. TO LPRINT
 A CKTYPE LISTING,  USE THE DIRE
CT COMMAND: RAND USR32086"
170 PAUSE 32768
180 FOR N=1 TO 682
190 POKE N+32085,CODE A$(N)
200 NEXT N
210 NEW
```

FOR YOUR QL

ARCHIVE SECRETS

by Real Gagnon

ARCHIVE is a powerful database program but there are some little known secrets that can make life easier to every programmer.

ARCHIVE version 2.35 introduced a whole new control characters set which are undocummented in the ARCHIVE reference manual. These characters adds more options to the ARCHIVE PRINT command. A complete list can be found in the ARCHIVE RUN-TIME manual only available from PSION or maybe from a few QL dealer.

We learn that some characters with ASCII code inferior to 32 get special attention from the ARCHIVE screen driver.

Some of these characters have their equivalent in ARCHIVE language, for exemple, take CHR(1), it can be used to set the INK color, the form is CHR(1)+CHR(n) where n is the color number (same number as in SUPERBASIC).

PRINT "This a"+CHR(1)+CHR(2)+"test" is equivalent to
PRINT "This a ";ink 2;"test" .

But some of these codes have a more unique effect, let's see some of them.

CHR(4)+CHR(c)+CHR(r)	is very useful, CHR(c) will be repeated "r" times. If we have PRINT CHR(4)+"*"+chr(80), the character "*" will be displayed 80 times on the screen.
CHR(5)	is an underline switch. Try this: PRINT CHR(5)+"SINCLAIR"+CHR(5)+" QL"
CHR(6)	moves the cursor to the right
CHR(8)	moves the cursor to the left
CHR(9)+CHR(c)	Same as SUPERBASIC PRINT TO c, where c is the column number.
CHR(10)	moves the cursor down
CHR(11)	moves the cursor up

CHR(12) erases the screen like CLS.

CHR(14) then you see the cursor flashing

CHR(15) then you don't see the cursor flashing

CHR(18)+CHR(n) is the equivalent to the SUPERBASIC OVER command.
n=0 then PRINT with INK on current PAPER
n=1 then PRINT with INK with TRANSPARENT paper, it's overprinting.

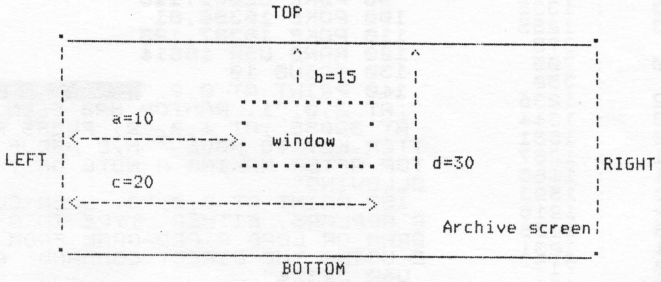
n=2 then PRINT with INK but XOR the data on the screen.

CHR(20)+CHR(a)+CHR(b)
+CHR(c)+CHR(d) creates a WINDOW.
The coordinates are in "characters".
"a" is the left margin from the left side of the screen.
"b" is the top margin from the top of the screen.
"c" is the right margin from the left side of the screen.
"d" is the bottom margin from the top of the screen.

An exemple will help to understand the ARCHIVE WINDOW creation.

PRINT CHR(20)+CHR(10)+CHR(15)+CHR(20)+
CHR(30)

will create the following WINDOW.



There is only one active WINDOW at one time.
The initial ARCHIVE WINDOW is defined with the following parameters:
CHR(20)+CHR(0)+CHR(0)+CHR(80)+CHR(25)

CHR(21)+CHR(n) scrolls up "n" lines.
CHR(22)+CHR(n) scrolls down "n" lines.
CHR(23)+CHR(n) scrolls left "c" columns.
CHR(24)+CHR(n) scrolls right "c" columns.

CHR(26) exchanges the PAPER with the INK, it's INVERSE printing.

CHR(27)+CHR(n) is for special CLS.
n=65 , CLS from the cursor position to the end of line.
n=66 , CLS from the cursor position to the end of the screen/window.
n=67 , STORE the cursor position.
n=68 , RESTORE the cursor position.

CHR(30) puts the cursor at the position 0,0 without erasing the screen.

CHR(31)+CHR(a)+CHR(b) Same as AT a,b . PRINT at line "a", column "b".

There is more control codes, but these are the most useful ones.

A very useful undocummented feature with the SEDIT command is hidden in ARCHIVE version 2.35.

When designing a screen, ARCHIVE provides some graphic characters to draw boxes. To use them, in SEDIT, do F5 and after press any key between "a" and "k". When drawing a box, you don't have to press each time F5 and the right key because SEDIT gives an easy way to repeat the last typed characters, just keep your finger on SHIFT and press a cursor key.

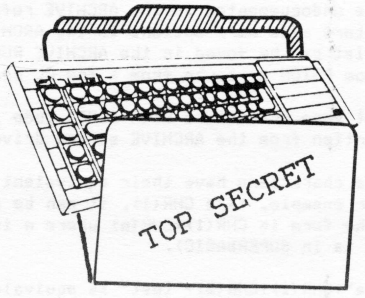
These graphic characters are similar to those found on IBM GRAPHICS compatible printer. Run the following program to print a reference sheet for the 11 new characters with the corresponding ARCHIVE codes. Your printer must be set for the IBM GRAPHICS character set #2, but anyway if your printer do not have the capability to print them, you can still use them in ARCHIVE!

SuperBasic program to print a
reference sheet for ARCHIVE
graphic characters.

```
 1 REMark by Real Gagnon  Montreal May 1988
 2 :
10 OPEN #4,ser1
20 :
30 PRINT #4,'.ARCHIVE graphic characters (SEDIT)
35 PRINT #4,' function key F5 then one of the following keys:'
37 PRINT #4,CHR$(27);'A';CHR$(18) : REMark set line spacing (opt.)
40 PRINT #4,'KEY   ','   CHARACTER','   ARCHIVE code'
45 RESTORE
47 :
50 REPeat loop
60   READ sed,ibm,ql$
70   PRINT#4, CHR$(sed),,CHR$(ibm),,ql$
80   IF sed=107:EXIT loop
90 END REPeat loop
95 :
100 DATA 97,179,'224' :REMark a
110 DATA 98,180,'225' :REMark b
120 DATA 99,191,'226' :REMark c
130 DATA 100,192,'227':REMark d
140 DATA 101,193,'228':REMark e
150 DATA 102,194,'229':REMark f
160 DATA 103,195,'230':REMark g
170 DATA 104,196,'231':REMark h
180 DATA 105,197,'232':REMark i
190 DATA 106,217,'233':REMark j
200 DATA 107,218,'234':REMark k
```

.ARCHIVE GRAPHIC CHARACTERS (SEDIT)
FUNCTION KEY F5 THEN ONE OF THE FOLLOWING KEYS:

KEY	CHARACTER	ARCHIVE CODE
A	│	224
B	┤	225
C	┐	226
D	└	227
E	┴	228
F	┬	229
G	├	230
H	─	231
I	┼	232
J	┘	233
K	┌	234



PC-IMPORT

How Does This IBM Translator Software Package From MINNY ELECTRONICS Stack Up??

reviewed by Michael E. Carver

This package bills itself as "a file transfer and BASIC language dialect translater". PC-Import will allow the user, who has access to an IBM Personal Computer (or compatible) equipped with a communication board and a QL, to "download" and translate MicroSoft BASIC programs into SuperBASIC, which will run independently on the QL. After a couple of false starts, I was able to successfully import IBM BASIC programs into my QL.

The first problem was hardware oriented. The manual assumes the user has a ready-made RS-232 cable. As I built my own cable, there were some problems not addressed in the accompanying manual. The QL serial ports only have 5 lines (GND, Txd, RxD, DTR, and CTS). The IBM serial ports have 9 different lines. In order for IBM BIOS to send messages out the serial port, both the DSR and CTS lines must be "true". I had to "tie" the DSR and CTS pins on the IBM together, to achieve communications from the IBM to the QL. This was accomplished by soldering a "jumper" wire between pins 5 and 6 (CTS & DTR) on the IBM cable end.

The other false start was due to an error in the manual. The manual provides step-by-step instructions on preparing MS-DOS to send information out its RS-232 port. To accomplish this, the printer output (LPT1) is directed to the communication output (COM1). The instructions successfully got me to this point, but the steps to direct a copy of the BASIC program to the printer output was in error. "COPY A: [file_name} LST1" would not work. This should have read, "COPY A: {file_name} LPT1". With this minor correction, I received data from the IBM successfully every time.

Before a BASIC program is transferred to the QL, it must be in ASCII format. The manual provides sufficient instruction on how to insure the BASIC files are in ASCII. As the program is "sent" to the QL, it is written to a file on a user-specified microdrive. The transfer portion of PC-IMPORT is straightforward and quick. The transfer rate is 4800 baud and only takes seconds.

The most important part of the program is the translation of IBM's BASIC into SuperBASIC. As PC-IMPORT is written entirely in BASIC, the translation is slow. Approximately 2.8 bytes are translated per second. A 4551 byte program took 27 minutes and 16 seconds to translate. The manual claims that up to 90 percent of the translation work can be achieved by PC-IMPORT. I found this figure to be highly overstated. Only a portion of non-compatible MicroSoft BASIC is translated into SuperBASIC (see Table 1). A fair knowledge of BASIC programming, and an ability to follow a BASIC listing to fathom the flow and logic are required to successfully get most imported BASIC programs to run on the QL.

As an acid-test, I took a text-manipulation program I had written on an IBM at work as an example. The program includes no graphics and simply accepts input from the keyboard. It then breaks up the text into proper spacing for book card labels. Due to major differences between the way the IBM BASIC handles string manipulation and undefined variables, I ran into many problems getting the program to run on the QL. Even though I had written the program, it was hard to follow the logic and correctly make the "hand-translations" required. One of the major problems was caused by the incomplete PC-IMPORT translation of the MID$ command.

A 2048 byte program (translated in 12:09 minutes) took me about 20 to 25 minutes to re-edit and hand-translate to achieve proper execution. I would not have been able to achieve this without the constant referral to the MicroSoft BASIC manual. I have had limited experience in BASIC programming on an IBM. (I would like to take a second to applaud the Sinclair BASICs. They are far easier languages to program, than MicroSoft BASIC.) If you do not have access to a MicroSoft manual, check your local library, you'll probably need it. As SuperBASIC will mark any BASIC line with "MISTAKE", the "hand-translation" job is made easier. The PC-IMPORT manual does provide a small table of BASICA commands with SuperBASIC equivalents, if any.

Most of the programs I imported dealt with graphics, but none of the graphic commands are translated by PC-IMPORT. One thing to keep in mind when translating most BASICs into Sinclair BASIC, is that their graphic screens are upside down (point 0,0 is in the upper-left-hand corner). Many of the IBM graphic commands can be imitated by creating procedures with SuperBASIC, to achieve the proper results (see Listing 1).

Some other uses for PC-IMPORT, which aren't mentioned in the manual, include: downloading BASIC ASCII files from bulleting boards and translating them into SuperBASIC. Using QUILL to type in a BASIC listing, printing it to a microdrive file and using PC-IMPORT to translate. The printer drive with QUILL

SHARP'S IS THE LARGEST QL SOFTWARE AND HARDWARE DEALER!

WRITE FOR FREE CATALOG

Sharp's, Inc.

VISA

Rt. 10, Box 459
Mechanicsville, VA 23111
(804) 746-1664 or 730-9697

MasterCard

will need to be altered to print a useable file to the microdrive (see Table 2). One may also use a simple program to enter an IBM listing directly into a microdrive file (see Listing 2).

I have mixed feelings about PC-IMPORT. It is slow and incomplete. PC-IMPORT must be viewed only as a programmer's tool. To achieve full usefulness from PC-IMPORT, the user must be a fair to accomplished BASIC programmer. If one has access to an IBM (or any other "on-line" source of MicroSoft BASIC), a lot of typing and raw translating can be avoided. I believe that PC-IMPORT could have been a much better program had other IBM BASIC commands been supported (i.e., PSET, INSTR, SPACE$, "\" or Interger Division, LOG). Also commands such as MID$ should have been fully translated, or RANDOMIZE should have been translated to the British spelling. I only hope that Minny Electronics will provide updated versions of this product in the future.

This program was obtained for review from: RMG Enterprises, 1419 1/2 7th Street, Oregon City, OR 97045, (503) 655-7484.

Table 1 -- IBM commands translated into Super BASIC

INPUT	DATA	THEN
GOTO	GOSUB	TAB
TO	'	ASC
COLOR	SQR	SGN
LOCATE	LEFT$	MID$
RIGHT$	STRING$	ELSE
VAL		

Table 2 -- Quill Printer Driver for ASCII BASIC

DRIVER NAME	:BASIC
PORT	:ser1
BAUD RATE	:9600
PARITY	:NONE
LINES/PAGE	:255
CHARACTERS/LINE	:255
CONTINUOUS FORMS	:YES
END OF LINE CODE	:LF
PREAMBLE CODE	:NONE
POSTAMBLE CODE	:NONE
All other options	:NONE

NOTE: To send the ASCII BASIC to microdrive, the above printer driver must be installed or be present on the default drive (usually #1) as "PRINTER_DAT". Select the Print option from within Quill and direct output to [Device_filename] instead of the printer.

LISTING 1

```
1450 REMark ---- MicroSoft BASIC command to draw a box
1452 REMark ---- The two co-ords are adjacent corners of the box
1454 REMark ---- followed by ink color -- B = box or BF = Fill
1460 REMark ---- LINE (IX1,IY1)-(IX2,IY2),RND*2+1,BF
1462 :
1464 REMark ---- SuperBASIC translation using PROCEDURE box
1466 box IX1,IY1,IX2,IY2,RND(1 TO 7),1
1468 :
1470 REMark ---- LINE (IX1,IY1)-(IX2,IY2),0,B
1472 box IX1,IY1,IX2,IY2,0,0: REMark SuperBASIC translation
1474 :
9000 DEFine PROCedure box (x,y,xc,yc,crayon,all)
9005 FILL all: INK crayon
9010 LINE x,y TO xc,y TO xc,yc TO x,yc TO x,y
9015 FILL 0
```

MANDELBROT
-- A Fractal World
UPDATE

by Michael E. Carver

As I was developing the mandelbrot program, I was unable to drive my monitor in F1 mode. I was unaware of the difference between the height of the characters as sent to the Screen. The following listings will allow complete viewing of the mini_menu area and the "canvas" while the Madelbrot Sets are being drawn (when the QL is in F1-Monitor mode).

Key-in and run the following Listing. This contains a short machine code routine to send only 8 lines of pixel information for each character (as opposed to 10 in Monitor mode). The program will self-install into the machine and save to MDV1_.

```
10 REMark **** loader for SD.YINC assign
20 a=RESPR(68): RESTORE
30 FOR x=0 TO 67 STEP 2
40   READ num: POKE_W (a+x),num
50 END FOR x
60 SBYTES mdv1_YINC_code,a,68
1000 DATA 17914,56,8316,1,1,28681,30463,20035
1010 DATA 8316,2,2,28681,30463,20035,28680,9326
1020 DATA 48,-16132,40,-10816,8310,-22528,17914,12
1030 DATA 28681,30463,20035,28672,20085,12668,8,40
1040 DATA 28672,20085
```


THE MOST COMPREHENSIVE RESOURCE FOR THE SINCLAIR QL. GET YOUR COPY TODAY!!

More and more QL owners are discovering the excellent new book by Mike de Sosa — **TAKING THE QUANTUM LEAP: The Last Word On The Sinclair QL.**

This 280 page book is chock full of useful programs and original programming examples. Chapters on using the bundled software and a look at the latest hardware and software releases. Written for both the novice and more advanced users. Priced less than most software packages and nearly two pounds of information!!

$26 (USA).

Exclusively available from:

TIME DESIGNS

29772 Hult Road, Colton, Oregon, 97017, USA. Telephone (503) 824-2658.

VISA and MASTERCARD accepted.

For a sample QL and Spectrum magazine, send $3

Insure that your Master Mandelbrot cartridge contains a copy of the newly created code (YINC_code). Insert (or merge) the following listing into the BASIC listing of the Mandelbrot program:

```
1905  CALL yinc
2555 yinc=RESPR(68): LBYTES mdv1_yinc_code,yinc: CALL
yinc
2905  CALL yinc
3755  CALL yinc
4605  CALL yinc
4665  CALL yinc
```

Time Designs Tests

QRAM, Archivist MP, Text87, & Mailbag

by Mike de Sosa

QJump's QRAM v. 1.16 * * * * 1/2

Earlier versions of Tony Tebby's **QRAM**, tested as part of the **Sandy SuperQBoard** system, troubled me, but now I realize that this was mainly due to flaws in the system and not in the **QRAM** software, itself. Then there were the spurious rumors that Tony had designed QRAM to be incompatible with the software of some of his competitors (**Supercharge** and **QLiberator**, to name two). Whatever the whole truth, much is now improved. **QRAM** now seems to work well with **SPEED-SCREEN**, **Trump Card** and other disk interfaces and RAM packs, but not with **FLASHBACK** or **TurboQuill+**--two top-notch programs.

QRAM comprises RAM-based utilities for the QL: a full multitasking front end, pop-up menus (within your Psion programs), fast and versatile RAMdisk software, dual-keystroke hotkeys, screen and window dumps, spoolers, a good compatibility with Tebby's **QDOS** and **SUPERTOOLKIT II** and other software, and many other utilities.

QRAM, already very efficient as it comes, offers many opportunities for customization. In its "stock" configuration, it "comes up" in the standard QL dual-screen format. Keying **Alt** / from the **SuperBASIC** format or within Psion or other programs presents the initial pop-up menu with six main options (FILES, JOBS, CHANNELS, PRINT, WINDOW DUMP, OPTIONS, and several redundant controls for QUIT, HELP, and moving the menu window. Two methods of selecting options are always available: by means of a pointer controllable with the cursor or **QIMI** (QL Internal Mouse Interface) and by keying the first letter of an option. The SPACE bar is usually used to select an action or file, and ENTER to execute a selected command.

QRAM is most economical in the use of keystrokes required to get into, out of, and between programs, especially if one customizes the BOOT and HOTKEY files. As many runs of one or more programs can be multitasked and switched between as memory permits and this capability is enhanced by **Grabber**, a utility which modifies programs such as Psion's **QL QUILL** so that they don't gobble up all available memory. A fraction of a star is deducted for its sometimes difficult or skimpy documentation and because it is not, at least in this version, compatible with some important new programs. If **TASKMASTER** is the Mercedes Benz of full-featured multitaskers, **QRAM** is the BMW, handier and superior in some ways.

HOT TIPS: Grabber-modified Psion programs and RAMdisk software (RAMprt) is usable separately without **QRAM**.
About $55, but make sure it's version 1.16!

A.R.K.'s ARCHIVIST MP * * * * 1/2

Applied Research Kernel Distribution's **ARCHIVIST MP** database manager is not an upgrade of **ARCHIVIST 128**, but a quantum leap forward. MP is for Multi-file Programmable: it can open and use up to 20 files at a time and may be programmed at several levels and in many ways to meet special requirements--it can access up to 200,000 records (over 100,000 with **Trump Card**). (It is also compatible with **ARCHIVIST 128** files and screen formats and can use **QL ARCHIVE** database, screen, and export files.)

Making use of **Run-time ARCHIVE**, it runs alone but is programmable using **QL ARCHIVE v. 2.38** (**ARCHDEV**). A control file option is available by which one can set up a multi-file system and automatically copy selected database and screen files to RAMdisk at turn on, enabling rapid access to data. Entering the name of a single control file can initiate use of a comprehensive system of database files and multiple screen formats. Global searches of various types can be undertaken among open files.

To simplify matters, **ARCHIVIST MP** makes use of a standard 22 renamable fields of up to 69 characters, one field per line. This arrangement is extremely practical--most database designers fail to "keep it simple, stupid," and create their own pitfalls.

ARCHIVIST MP is in many ways comparable to high-capacity database handlers like **dBase 3**: rational menu structures offer single keystroke access to various functions and "external" utility programs can be executed from a "Tools Menu"--the example program supplied has many useful utilities including one to sum values in a given field throughout a file and those to globally delete, insert, or replace strings of text in any field of every record. Documentation is excellent and a tutorial on database use.

A fraction of a star was deducted because all **ARCHIVIST MP** fields are string fields, making it more difficult to compute and insert the value of interactive mathematical values than it is using **QL ARCHIVE**, and because data is not directly transferable to external programs as it is using **FLASHBACK**.

Requiring a minimum of 256K total RAM, **ARCHIVIST MP** is supplied on Microdrive cartridges and 3.5" or 5.25" disks. Available for about $76 ($56 if you return the original **ARCHIVIST 128** cartridge), using UK checks, Eurocheque, International GIRO, or VISA card, directly from ARK Distribution, Corve Farmhouse, Chale Green, Ventnor, PO38 2LA, U.K. (Telephone 0983 79 496). ARK will also supply **QL ARCHIVE v. 2.38**, for about $36, and other software.

Software87's Text^{87} v. 1.06 * * * * 1/2

Digital Precision's excellent text editor **THE EDITOR** was subtitled "Chuck Quill Out!" by its designers, but even the advanced **Special Edition** of **THE EDITOR** failed to oust **QL QUILL** from its WYSIWYG ("What you see is what you get") supremacy, especially after such great QUILL-enhancers as **SPEEDSCREEN**, **TurboQuill+**, and **FLASHBACK** greatly increased the performance of **QL QUILL** without decreasing its user-friendliness.

Text^{87} promises, repeat, promises to do just that, but it has a little way to go before it will supplant the cheapest and easiest of them all.

Text^{87} is billed as a full-featured WYSIWYG word processor designed for easy use by amateurs. The printer driver permits configuration for "any type of printer," but, if yours is not one of the several preprogrammed types, it is not directly configurable without an assembler.

Text^{87} is menu-extensive and typeface- and typesize intensive: it will print anything that your printer can--assuming compatibility--and offers comprehensive functions, including easy block transfer to other files, **QL QUILL's** main drawback. It also loads **QL QUILL** _doc files.

Text^{87} is very compact (64K) and requires a minimum of 64K RAM expansion (a total of 192K RAM)--more is better. With 256K RAM, **Text^{87}** is said to load a 3400-word QUILL _doc file in 12 seconds and a 33000-word file in 52 seconds. Printers directly supported in this version include the Epson FX80, LX800, and DX100; the Silver-Reed EXP500; the Brother HR10 and HR15; and the Diablo 630. The FX80 printer driver seems to work well on my old Star **Delta 10** printer. An ASCII printer driver is also supplied for use with obsolete or nonstandard printers.

Text^{87} is compatible with **QRAM**, **TASKMASTER**, **SPEEDSCREEN**, **FLASHBACK**, and **SPELLBOUND**. **Founted^{87}**, by the same firm, is a compatible and easy to use font (or fount) editor; **Fountext88** is a graphic printer driver for **Text^{87}** that provides more than 20 printer typefaces in different sizes for Epson-compatible dot-matrix printers. **2488** is a set of dedicated printer drivers for Epson and NEC 24-pin printers that supports different print styles and sizes and proportionally spaced typefaces.

Text^{87} lies in complexity somewhere between **QL QUILL** and **The Editor**, perhaps closer to the latter. But, if you are ready and able to undertake learning some new word processor concepts, and have a compatible printer, and can afford it, **Text^{87}** should do about anything very quickly. It may represent a hybrid, rather futuristic combination between a word processor and a desktop publisher.

The documentation I have is in the form of a not always clear or complete 60-page manual; for example, not all commands and options found in the submenus are covered. But a revised manual, including a new tutorial is probably available now.

I had intended to do a full article on the capabilities and operation of **Text^{87}** for this issue, but late notification of a short deadline made this impossible. Next time for sure.

Text^{87} is now available directly from Software 87, 33 Savernake Road, London NW3 2JU, U.K. Airmailed prices for **Text^{87}** and **Founted^{87}** are $75 and $15, respectively. **Fountex^{88}** and **2488** are $45 and $15, respectively. Payment must be made by traveler's check, International GIRO postal money order, or other check directly payable at a U.K. bank; add about $8.50 for checks not so payable!

EMSOFT'S MAILBAG * * * * 1/4

MAILBAG is American software consisting of database handling programs and screens for use with **QL ARCHIVE**, a minimum of 256K RAM is required. It provides a versatile database for purposes ranging from a simple address to small-business use. Its designer Peter Hale of Boston is high on it, saying that it is "the most exciting program for ARCHIVE that has yet been released," distinguished from other database applications for the **QL** by its great flexibility in handling names and addresses, its lack of protection from illegal pirating, and in offering menu-driven **QL ARCHIVE** programs. It also offers mailmerge facilities using specially prepared **QL QUILL** documents.

Documentation consists of a 4-page flyer and a 12-page _doc file, but I understand that you may never have to use the latter. I did!

Peter claims that **MAILBAG** is unique in four respects:

a. It prints labels to a U.S. Postal Service standard

b. It is unprotected and may be fully user edited for screen displays, programs, and listings

c. It prints with versatility without having to change **printer_dat** files.

d. The user may design custom formats for record display on printouts or as mailmerge documents.

I didn't have a lot of time to really "wring it out" and I am not a database person, but from what I have seen it is a winner. I shaved part of a star because it should be on **runtime ARCHIVE** and stand alone--maybe in its next version.

Excellent work Peter!

MAILBAG is available directly from EMSOFT, Box 8763, Boston, Mass. 02114-8763 for $19.95 on 5 1/4" 80-track floppy disk or $21.95 on Microdrive cartridge. No credit cards. Dealer prices are available.

NEXT TIME: A complete and thorough treatise on **Text^{87}** and more hot software.

Z COLUMN

by Tim Woods

There is a lot to report on this issue, as Z88 activity keeps moving along. But first, an explanation is in order, for those joining us for the first time. The "Z-COLUMN" is a regular feature that discusses the newest member of the Sinclair computer family...the Z88 Laptop. While maintaining some of the characteristics of earlier Sinclair machines (Z80 CPU, very light-weight/compact design, use of function keys or combination of keys to execute major commands, etc.), the Z88 achieves a whole new standard of performance and power. If you haven't seen one or tried one out yet, there just might be a computer dealer in your area.

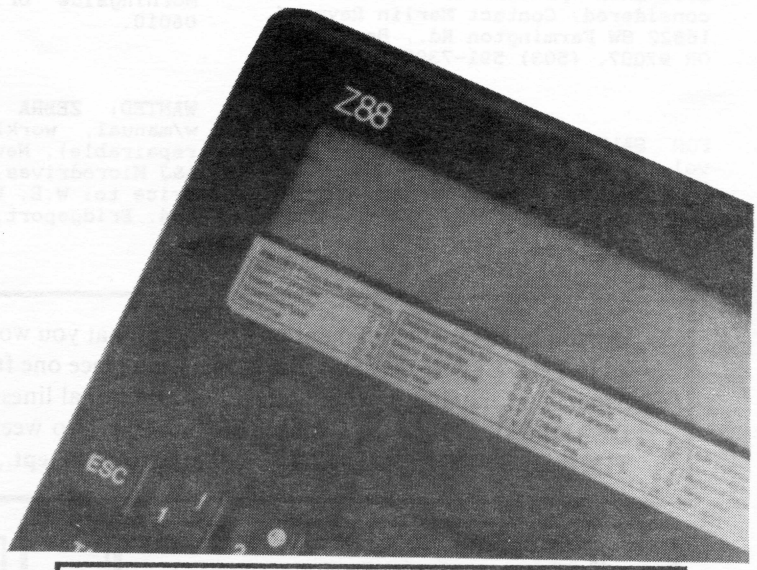
As of this writing, there are three organizations involved in marketing the Z88 in the U.S.A. First, there is SSI Computer Systems in Portland, Maine, who distributes the Z88 inventory from their New England warehouse. SSI is being advised by The Marketing Clinic, which is run by long time Sinclair associate, Nigel Searle, who is also the former head of Sinclair Research, U.S. division. And finally, there is another organization near Chicago, Illinois, called Cambridge Direct Ltd., operated by President, Kevin Jeffers. Cambridge Direct is the exclusive direct marketing arm for the Z88. To date, I feel that an adequate job is being conducted to pomote and market the Z88. There are several hundred dealers around the U.S. that are stocking them. A large ad recently appeared in the WASHINGTON POST newspaper, and very favorable reviews of the machine are in the most current issues of BYTE and COMPUTER SHOPPER.

The first issue of Z88 USER, "the official Z88 computer magazine" hit the streets in April. The 34 page publication is being produced by the publishers of QL WORLD Magazine. It looks like it will be a good magazine as soon as they get rolling along on a regular schedule. A question and answer column covered two items that are undocumented in the Z88 User Guide.

Another publication, which I am personally involved in is called CLUB Z88. It is a non-profit bi-monthly newsletter. One unique feature, is that subscribers can earn points, free gifts, and a chance to sit on the exclusive "board of directors" by contributing articles, tips, and programs to the newsletter. You can obtain a sample issue, by sending a SASE to: CLUB Z88, c/o Time Designs, 29722 Hult Rd., Colton, OR 97017.

New products are just now coming on the market: a spelling checker, the official Cambridge 300/1200 "matchbook" modem, A "C" Compiler, the Advanced User Guide, a cassette tape interface to provide security back up for the RAM cards, and the elusive one meg RAM cards are still being promised before the year's end.

As a tip for PIPEDREAM (the on-board word processor): I have found that following the guidlines specified for moving the cursor position around, using TAB, arrow keys, diamond key, etc., that getting into the habit of using these will develop faster manipulation of your text. I know this sounds like common sense, but more often than not, we use old typing habits in word processors that can really slow things down.



THE Z88
IS HERE!
And we have it!

Sir Clive's LATEST is now in stock at RMG!! This SUPER little laptop computer with features like:
128K RAM, 32K ROM, BUILT-IN S/W PACKAGES AND SMALL SIZE AND WEIGHT (2 LBS!)
make it a GREAT addition to our line. As our way of introducing you to the Z88, we are offering, for a limited time only, with any Z88 purchased before 9/30/88,
A FREE CENT. PAR. I/F
A 32K RAM CARTRIDGE

THE PRICE? TOO LOW TO LIST
PLEASE CALL OR WRITE!

RMG ENTERPRISES
1419 1/2 7TH STREET
OREGON CITY, OREGON 97045
503/655-7484 * NOON-10 TUE-SAT

The Classified

FREE ADS FOR SUBSCRIBERS

FOR SALE: EYE-Q, retail $50, sell for $35, or best offer. MATCHPOINT, retail $28, sell for $18, or best offer. Shipping included. WANTED: TECHNI_QL. Chia-Chi Chao, 73 Sullivan Dr., Moraga, CA 94556.

WANTED: Z-TALKER FOR TS1000. Please contact Merlin J. Raymond, 16822 Farmington Rd., Beaverton, OR 97007.

WANTED: I/O PORTS FOR TS1000. BYTE-BACK preferred, but others considered. Contact Merlin Raymond, 16822 SW Farmington Rd., Beaverton, OR 97007, (503) 591-7392.

FOR SALE: bound copies of SYNTAX vol.1.1 to vol.5.1 plus three SQ Quarterlies. $40.00 or free to non-profit. Peter Hale, Box 8763, Boston, MA 02114.

WANTED: Booklet on 2050 Modem and any relevant software. Also full-sized keyboard. N. Oshana, 187 Morningside Dr. E., Bristol, CT 06010.

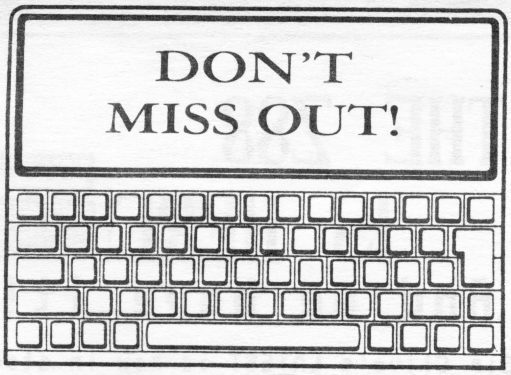
WANTED: ZEBRA GRAPHICS TABLET, w/manual, working or defective (if repairable). New or used wafers for A&J Microdrives. Send description & price to: W.E. Powden Sr., R#1 Box 364, Bridgeport, IL 62417.

FORTH 79,83 USERS, I would like you to drop me a line to swap information and help. George David Johnson, Beaufort Naval Hospital, PO Box 6204-A, Beaufort, SC 29902.

BOOK WANTED: "S.U.M.-Small User's Math- Powerful Algorithms". Willing to pay any reasonable price. Jaime A Cruz-Figueroa, Rt 2 Box 245-M, Lillington, NC 27546.

WILL TRADE: PRO/FILE 2068 for ZX PRO/FILE. Also trade BYTE-BACK MD-68 for MD-2B, or sell out-right (make offer). D.G. Smith, R.415 Stone St., Johnstown, PA 15906.

Do you have some equipment or a program that you would like to sell? Looking for something hard to find? Place an ad in **THE CLASSIFIEDS!** Subscribers can place one free personal ad in each issue. Ad size is 32 Col. wide (like 2040 paper) and maximum of six lines. For additional lines - $3 each. **NON-SUBSCRIBERS** and **DEALERS:** $4 a line. **DEADLINE FOR ALL CLASSIFIED ADS:** Two weeks before publication date. Mail your ad to: **TIME DESIGNS MAGAZINE, The Classifieds Dept., 29722 Hult Rd., Colton, Oregon 97017.**



Subscribe Today

only $16.95
year

TIME DESIGNS MAGAZINE CO.
29722 Hult Rd., Colton, OR 97017

☐ New subscription ☐ Renewal

Name _____

Address _____

City _____

State _____ Zip _____

SPECTERM-64 (TS-4.1)
1200 Baud Terminal Software For The 2068
On JLO, AERCO, Larken disk or tape
and
Z-SI/O Bare Boards
RS-232 Serial Interface For The 2068
With FREE 2050 card
both for only
$50.00 + $3.50 S&H
(Outside 48 contiguous states, $6.50 S&H)

This is a TDM special offer. To order or for additional info contact:

Ed Grey Enterprises
P.O. Box #2186, Inglewood CA 90305
(213) 759-7406

The Grey Matter BBS & RCP/M
(213) 971-6260

APPLICATIONS SOFTWARE for the QL

MAILBAG An address database you won't believe. Labels, Rotary index cards, Zip sorting, tickler filing 3 column printout, many **automatic features**, not protected 256K+

TAX-I-QL Spreadsheet for **IRS** tax returns 384K+

QLANDLORD Manages up to 99 units or buildings and does Schedule E (Soon) 256K+

$19.95 ea ppd on 5 1/4" DSQD
(+$2.00 on microcartridge)

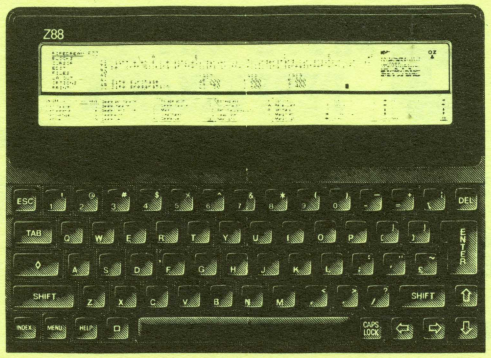
Agents for ZX/TS **Siriusware**
4K Wordprocessor with
TS 2040 lower case printer driver

Orders, Catalogues & Dealer info:

EMSOFT
P.O. Box 8763, Boston, MA 02114
(617) 889-0830

THE Z88 UNDER 2 LBS.

A Computer Without Compromise



• Where laptops compromise on display and RAM capacity to achieve portability, and desktops seem to equate price with power, the Z88 is a personal computer which makes no compromises • A CMOS-technology computer with the power to address 4 Mbytes of memory • A computer with a work-free display of 8 lines of 80 characters, an LCD screen which outdates all others, and a unique dynamic page map on screen • A computer with solid-state permanent storage • A computer with advanced word-processing, spreadsheet and ingenious time- and data-management software built-in • A computer which is completely self-contained, which gives you up to 20 hours active computing from just 4 AA batteries, yet which talks and listens to your IBM • A computer with a full-size keyboard, in a package less than the size of an 8½x11, with a total weight of less than 2 lbs. • The Z88. A computer without compromise.

WRITE FOR FREE CATALOG

Sharp's, Inc.
Rt. 10, Box 459
Mechanicsville, VA 23111
(804) 746-1664 or 730-9697

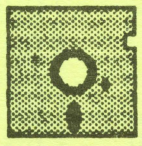
LET YOUR IMAGINATION SOAR!

Be sure to stop by our booth at
THE 3RD ANNUAL INTERNATIONAL/
GREAT NW TS MINI-FAIR!

SAVE!

Valuable Coupon!
$5 OFF
Any order for $50 or more when you include a copy of this coupon. Use at the Fair or anytime until 11/1/88.

LARKEN SYSTEMS FEATURED HERE!



The LKDOS disk I/F and EXTENDED BASIC CARTRIDGE and the NEW LARKEN RAMDISK will be on display at RMG's booth and you are invited to stop by to see these GREAT PRODUCTS demoed!

MINI-FAIR SUPER BUY!
Order any COMPLETE LKDOS SYSTEM and get the LARKEN DISK EDITOR and 2 other disk packages AT ABSOLUTELY NO EXTRA CHARGE!
-OR-
Order the LKDOS system WITH a RAMDISK and we will throw in 1 FREE 32K RAM CHIP! (Either way, you get over $20 in FREE items!)

MasterCard

RMG ENTERPRISES
1419 1/2 7TH STREET * OREGON CITY, OR 97045 * 503/655-7484

VISA

(Lemke Software Development 2144 White Oak Wichita, KS 67207)

(Lemke Software Development 2144 White Oak Wichita, KS 67207)

(Lemke Software Development 2144 White Oak Wichita, KS 67207)

Pixel Print **PLUS!**

THE DESKTOP PUBLISHER
by Lemke Software

What's the PLUS?

PERFORMANCE!

Checkout these SPECS:

1) WYSIWYG (What You See Is What You Get!)
Create your text on the screen... as easy as typing! LOAD graphics... Load ICONs... Import TASWORD text via the TASWORD conversion utility.

2) AUTOMATIC and MANUAL line and character adjustments.

3) RESTORE FONT (after using the BOLD/MODERN/ITALIC modifiers.

4) KEEP/UNDO/SAVE/LOAD/LOAD ICON
LOAD SCREEN$/SAVE SCREEN$
WIDE/HIGH/CLS/SCROLL SPEED

5) OVER/INVERSE/CAPS LOCK
UP TO 16 POINT FONTS
(font package in development)

6) COPY/ERASE/INSERT/DELETE/NEW

7) AERCO/TASMAN A&B/A&J CPI

8) IBM/EPSON/PROWRITER type printers.

Pixel Print **PLUS!** has all of the features found in **v2.8**, it is 100% compatible with the Pixel Print **ICON** #1 & #2 packages, **FONTs** #1, #2, & #3 as well as the **TASWORD Util.** and your Pixel Print files!!

PIXEL PRINT PLUS!....	$19.95
TASWORD TEXT CONV....	$19.95
ICON PACKAGE #1.....	$19.95
ICON LIBRARY #2.....	$14.95
FONT PACKAGE #1.....	$19.95
FONT LIBRARY #2.....	$14.95
FONT LIBRARY #3.....	$19.95
16 POINT FONT DESIGNER	$14.95
PIXEL SKETCH v2.....	$19.95
CHECKBOOK BUDGET MASTER.....	$19.95

Pixel Print PLUS!...

The **PLUS!** is 10 more functions to make the PIXEL PRINT DTP an even more powerful, easy to use program!!

LEMKE SOFTWARE DEVELOPMENT

Lemke Software
2144 White Oak
Wichita, Ks.
67207

Pixel Print Professional
AERCO DISK VERSION!

* Pixel Print Plus v3.2
* Create up to 20 page documents
* Print a complete Doc header, left, & right (no more printer adjustments!)
* Uses bank-switching to print left and right columns together
* Print multiple Copies!
* Newsletter Format or
* 64 Column Letter Format

NOTE: printers must be capable of 72 to 90 Dots per inch (576 to 720 dots per line) Bit Image Graphics!!!

* Automated Printer Customizing
* 5500 diskette (OR AUTO-SAVE TAPE)
* only $29.95 ppd. (ORDER NOW!)

OLIGER DISK VERSION!

* Same as the AERCO version above except for use with the **Oliger Disk Interface**
* Requires a minimum of 32K RAM expansion... (see the 32K RAM Cartridge below!)
* Program comes on CASSETTE and AUTO-SAVES to DISK, some editing required (remove REMs)
* only $29.95 ppd. (ORDER NOW!)

Pixel Print Professional

For CASSETTE & MICRODRIVE

* Same as Aerco except for use with Tapes...

NOTE: Specify either TAPE or MICRODRIVE...

* Requires a minimum of 32K RAM expansion... (see the 32K RAM Cartridge below!)
* only $29.95 ppd. (ORDER NOW!)

32K RAM CARTRIDGE

* 32K Volatile Memory
* Fits into the TS-2068 Cartridge Dock

only $40.00 plus $4 shipping and handling

ONLY 6 LEFT... UNLESS WE SEE MORE INTEREST, THERE WON'T BE ANY MORE MADE!!

* RAM Cartridge is accessed via BANK-SWITCHING... and is NOT limited to use with the PIXEL PRINT PROFESSIONAL!

Watch for other Lemke Software to use this RAM expansion!!!